Planificación Dinámica de Planes de Ocio

Olétrip

MEMORIA DEL TRABAJO FIN DE GRADO

Realizado por
Raquel Álvarez Hernández
Yanyan Cheng
Qiang Sun

Dirigido por
Juan Antonio Recio García
Antonio Sánchez Ruiz-Granados

Grado de Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid
Junio 2016

Autorización de difusión

Raquel Álvarez Hernández, **Yanyan Cheng** y **Qiang Sun**, y por otra parte, **Juan Antonio Recio García** y **Antonio Sánchez Ruiz-Granados**, como alumnos/as y tutores, del Trabajo de fin de Grado (TFG) del Grado de Ingeniería Informática de la Facultad de Ingeniería Informática, autorizan, mediante el presente documento, a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a los autores del mismo, cuyos datos se verán reflejados a continuación. De igual forma, autorizan a la Universidad Complutense de Madrid a que sea depositado en acceso abierto en el repositorio institucional con el objeto de incrementar la difusión, uso e impacto del TFG en Internet y garantizar su preservación y acceso a largo plazo

Planificación Dinámica de Planes de Ocio: Olétrip

Curso Académico 2015-2016

Alumnos

Raquel Álvarez Hernández

Yanyan Cheng

Qiang sun

Tutores

Juan Antonio Recio

Antonio Sánchez Ruiz-Granados

Firma de los alumnos ***Firma de los tutores***

Si buscas resultados distintos, no hagas siempre lo mismo

Albert Einstein

Agradecimientos

En primer lugar y más importante, queremos agradecer a nuestros padres, por estar siempre ahí, por querernos, apoyarnos, ayudarnos, aconsejarnos y darnos todo lo impensable siempre. Sin vosotros nada en nuestras vidas hubiese sido posible.

En cuanto al diseño de la base de datos, queremos agradecer por las contribuciones de conocimiento sobre este campo del profesor Manuel Montenegro Montes, estas ayudas han sido una clave importante en cuanto a la simplicidad y coherencia del esquema final, así como a la empresa que nos ha facilitado datos sobre lugares de la comunidad autónoma de Andalucía.

Les agradecemos a nuestros tutores, Juan Antonio Recio y Antonio Sánchez Ruiz-Granados por todo el tiempo, el esfuerzo y paciencia que han invertido en nosotros, y en este proyecto, por haber estado siempre ahí, resolviendo todas nuestras dudas y equivocaciones, sin ellos este proyecto no hubiese sido el mismo.

También hemos de recordar a todos los profesores/as, compañeros/as, y hermanos/as, que han estado en las diferentes etapas de esta larga carrera, que es la vida, que han aportado su granito de arena para llegar a donde hemos llegado, que nos han apoyado, y sobretodo creído en nosotros. Gracias por todos estos años en los que hemos aprendido y crecido juntos.

Y a todas las personas, que nos han ayudado a lo largo de todo el desarrollo de este trabajo, y demás proyectos de vida.

Resumen

En la actualidad, como consecuencia de los avances tecnológicos, hay una gran cantidad de sistemas de recomendación que proporcionan demasiada sobreinformación, y, sobre todo, los dedicados al sector turístico y de ocio, que cada vez están más en auge, y las personas recurren cada vez más a ellos para organizar planes, individuales o grupales. Sin embargo, sólo muestran información estática y no acorde con las preferencias de usuario, además de no poder ofrecer recomendaciones para grupos.

Olétrip surge como un sistema que permite la elaboración de planes de ocio individuales y/o grupales en la comunidad autónoma de Andalucía (España), basados en preferencias de usuario, y retroalimentación de todos los participantes en él. Llegando a conseguir el plan idóneo para todos.

Olétrip se ha diseñado e implementado mediante una serie de servicios REST, donde se recabarán datos de preferencias de viaje de los usuarios, retroalimentación de las diferentes actividades, etc., y así conseguir recomendar el plan deseado por el usuario.

Para probar el uso de estos servicios, se ha diseñado una interfaz web, independiente del sistema, implementada con tecnologías idóneas para tal fin, HTML y JavaScript, así como una evaluación con usuarios, para poder conocer el grado de usabilidad de la web, y los servicios que ofrece.

Además, se han explorado, analizado, y diseñado algoritmos que han permitido ofrecer una ruta final idónea.

Por lo tanto, se describirá como se ha llevado a cabo el proceso de elaboración del sistema, así como la implementación de los algoritmos, los resultados y conclusiones obtenidas tras todo el desarrollo.

Palabras clave

Ocio, recomendador grupal, servicios, interfaz, clúster, retroalimentación, planificador de viajes, tecnologías web, base de datos

Abstract

Nowadays, due to the age of rapid technological advance, there are a lot of different recommendation systems which provide too much unnecessary information. Most of those recommendation systems are dedicated to the tourism and leisure sector, which are booming, and are more and more frequently used to organize individual or group plans. However, they only show static information, and it is not in accordance with the user's preferences, moreover, they can't offer group recommendations.

Olétrip sets up as a system that allows elaborating individual and group leisure plans in the autonomous community of Andalusia (Spain), it utilizes the user's preferences and the feedback from all participants, in order to achieve the perfect plan for everyone.

Olétrip has been designed and implemented through series of REST services, where travel data will be collected (preferences of users, feedback from the different activities, etc.) in order to recommend the best custom plan for the user.

To test the use of these services, it has been designed a web interface, independent of the system, that has been implemented with suitable technologies for this purpose, HTML and JavaScript, as well as a final evaluation of this interface by the users, to determine the degree of usability of the web, and offered services.

In addition, it has been investigated, analyzed and designed algorithms that have allowed to offer a good final route.

Therefore, it will be described how the process of system development and implementation of algorithms has been carried out and the results and conclusions after the whole development.

Keywords

Leisure, group recommender, services, interface, cluster, feedback, trip planner, web technologies, database

ÍNDICE

AGRADECIMIENTOS ..... **V**

RESUMEN ..... **VII**

ABSTRACT ..... **IX**

1. INTRODUCCIÓN Y MOTIVACIÓN **1**

1.1. OBJETIVO 3
1.2. ESTRUCTURA DE LA MEMORIA 4

2. ESTADO DEL ARTE **5**

2.1. SISTEMAS DE RECOMENDACIÓN 5
2.2. SISTEMAS DE RECOMENDACIÓN RELACIONADOS CON EL OCIO Y TURISMO 12

3. TECNOLOGÍAS APLICADAS **19**

3.1. BACKEND – JAVA EE 19
3.2. FRONTEND – JAVASCRIPT 24

4. DESCRIPCIÓN FUNCIONAL **29**

4.1. EJEMPLO DE USO 29
4.2. CASOS DE USO 36

5. ARQUITECTURA DEL SISTEMA **49**

5.1. ARQUITECTURA MODULAR 49
5.2. BASE DE DATOS 51
5.3. SERVICIOS 55
5.4. INTERFAZ 59
5.5. MECANISMO DE RECOMENDACIÓN 71
5.6. MECANISMO DE CREACIÓN DE RUTA 73
5.7. MECANISMO DE RETROALIMENTACIÓN 83

6. EVALUACIÓN CON USUARIOS **87**

6.1. OBJETIVOS DE LA EVALUACIÓN 87
6.2. PARTICIPANTES 87
6.3. ENCUESTA INICIAL 87
6.4. MODERADOR 88
6.5. DESARROLLO DE LA EVALUACIÓN 88
6.6. ENCUESTA FINAL 89
6.7. RESULTADOS OBTENIDOS 89
6.8. MEJORAS CONCRETAS A REALIZAR 95

7. CONCLUSIONES Y VISIÓN FUTURA **99**

8. CONCLUSION AND FUTURE WORK **101**

9. APORTACIONES **103**

9.1. APORTACIONES DE RAQUEL ÁLVAREZ 103
9.2. APORTACIONES DE YANYAN CHENG 105

9.3. APORTACIONES DE QIANG SUN 107

APÉNDICES 109

A. BASE DE DATOS 111

B. SERVICIOS OLÉTRIP 115

C. MOCKUPS 137

D. CONFIGURACIÓN DEL ENTORNO 141

BIBLIOGRAFÍA 145

ÍNDICE DE FIGURAS

LOGO OLÉTRIP 1
SISTEMAS DE RECOMENDACIÓN 5
GUÍA REPSOL 14
TRIPADVISOR 15
MI NUBE 16
FEVER 17
TECNOLOGIAS USADAS 19
TRAZA LOG4J 23
TECNOLOGÍAS QUE FORMAN AJAX 25
SISTEMA RECOMENDADOR 29
EJEMPLO DE USO. INICIO 30
EJEMPLO DE USO. RECOMENDACIÓN 30
EJEMPLO DE USO. PREFERENCIAS 31
EJEMPLO DE USO. LUGARES 32
EJEMPLO DE USO. RUTA 33
EJEMPLO DE USO. INVITACIONES 34
EJEMPLO DE USO. DETALLES 35
EJEMPLO DE USO. ALTERNATIVA 35
EJEMPLO DE USO. FINALIZAR 36
DIAGRAMA DE CASOS DE USO 39
DIAGRAMA DE FLUJOS. USUARIO 46
DIAGRAMA DE FLUJOS. INVITACIONES 47
DIAGRAMA DE FLUJOS. RETROALIMENTACIÓN 47
ARQUITECTURA MODULAR 51
BASE DE DATOS. ESQUEMA 52
INTERFAZ. RESPONSIVE DESIGN 60
INTEFAZ. PÁGINA DE INICIO 62
INTEFAZ. PÁGINA DE DESCUBRE 63
INTERFAZ. REGISTRO 63
INTERFAZ. PÁGINA DE ACCESO 64
INTERFAZ. PREFERENCIAS DE VIAJE 65
INTERFAZ. SELECCIÓN DE LUGARES 66
INTERFAZ. PREVISUALIZACIÓN Y MODIFICACIÓN DE RUTA 67
INTERFAZ. PREVISUALIZACIÓN Y VOTACIONES DE RUTA 68
INTERFAZ. PERFIL DE USUARIO 69
INTERFAZ. RUTAS DE USUARIO 69
INTERFAZ. SUGERENCIA DE RUTAS 70
INTERFAZ. PREVISUALIZACION Y VOTACIONES (NO REGISTRADO) 71
K-MEANS. CLÚSTER 77
VOTACIÓN DE ACTIVIDADES 83
COMENTARIOS DE ACTIVIDADES 84
REEMPLAZO DE ACTIVIDADES 85
EVALUACIÓN CON USUARIOS. EDADES 89
EVALUACIÓN CON USUARIOS. NAVEGACIÓN 90

EVALUACIÓN CON USUARIOS. UBICACIÓN 90
EVALUACIÓN CON USUARIOS. PLATAFORMA 91
EVALUACIÓN CON USUARIOS. SEXO 91
EVALUACION DE USUARIOS. USO DE LA WEB 92
EVALUACIÓN CON USUARIOS. VIAJES EXISTENTES 92
EVALUACIÓN CON USUARIOS. FUNCIONALIDADES 92
EVALUACIÓN CON USUARIOS. PREFERENCIAS 93
EVALUACIÓN CON USUARIOS. CREACIÓN DE UN VIAJE 93
EVALUACIÓN CON USUARIOS. AÑADIR ACTIVIDADES 93
EVALUACIÓN CON USUARIOS. PLAN PROPUESTO 94
EVALUACIÓN CON USUARIOS. SUSTITUIR ACTIVIDADES 94
EVALUACIÓN CON USUARIOS. INVITAR AMIGOS 94
EVALUACIÓN. CAMBIO 1 97
MOCKUPS. PÁGINA DE INICIO 137
MOCKUPS. PÁGINA DE SUGERENCIAS 137
MOCKUPS. PÁGINA DE PREFERENCIAS 138
MOCKUPS. SELECCIÓN DE LUGARES 138
MOCKUPS. PREVISUALIZACIÓN DE VIAJE (CREADOR) 139
MOCKUPS. PREVISUALIZACIÓN DE VIAJE (INVITADOS) 139

ÍNDICE DE TABLAS

CASOS DE USO. ESCENARIO CREAR VIAJE 41
CASOS DE USO. RECOMENDAR PLANIFICACIÓN 42
CASOS DE USO. ESCENARIO INVITAR PARTICIPANTES 43
CASOS DE USO. ESCENARIO VOTAR ACTIVIDAD 44
CASOS DE USO. ESCENARIO COMENTAR ACTIVIDAD 46
K-MEANS 77
ALGORITMO VORAZ 82
BASE DE DATOS. USER 111
BASE DE DATOS. TRIP 112
BASE DE DATOS. OPINION 112
BASE DE DATOS. SITES 112
BASE DE DATOS. TRIP_USER 113
BASE DE DATOS. TRIP_SITES 113
BASE DE DATOS. TRIP_PREFERENCES 113
BASE DE DATOS. TRIP_CITIES 114
BASE DE DATOS. OPINION_COMMENT 114
CONFIGURACIÓN. MAVEN 141
CONFIGURACIÓN. JERSEY 142
CONFIGURACIÓN. LOG4J 143
CONFIGURACIÓN. PROPERTIES 143
CONFIGURACIÓN. MYSQL SERVER 143
CONFIGURACIÓN. TOMCAT 144

1. Introducción y motivación

Hoy en día, la mayoría de las personas que planean un viaje lo primero que hacen es iniciar una búsqueda a través de Internet. Cada vez más personas son conscientes de las ventajas de las nuevas tecnologías para la planificación de las actividades de ocio.

Sin embargo, los viajeros suelen tener un conocimiento limitado de la provincia/ciudad que quieren visitar y no son conscientes de los lugares artísticos, sociales, entretenimiento, etc., que estos pueden ofrecer, y sobre todo, cuando se habla de grandes comunidades, como lo es Andalucía.

Un usuario puede encontrar una gran cantidad de información sobre el destino deseado, invirtiendo mucho tiempo en la selección de las actividades que prefiere y en organizar adecuadamente la ruta deseada, a veces llegando a ser una tarea tediosa.

Olétrip analiza las diversas preferencias del usuario, y se seleccionan las actividades más interesantes para él. Como novedad, Olétrip tiene el objetivo de que, no solo una persona puede obtener un plan acorde con sus gustos, sino que un grupo de personas pueda hacer un plan y que todos los usuarios de ese grupo queden satisfechos con las actividades que se vayan a realizar.

En conclusión, Olétrip es un sistema que genera recomendaciones sobre recorridos turísticos en la Comunidad autónoma de Andalucía (España), la cual pretende que sea un servicio destinado a viajes individuales y/o grupales, generando los planes de actividades organizados teniendo en cuenta gustos y preferencias de los individuos.

1.1 LOGO OLÉTRIP

Cabe destacar, que Olétrip, está pensado para ser escalable, y poder añadir más lugares, y sobretodo, más comunidades, y así conseguir una aplicación que sea válida para más viajeros.

1.1. Objetivo

El objetivo que se pretende alcanzar con el presente trabajo es el desarrollo de un planificador dinámico de planes de ocio. Para poder lograr este objetivo, se pretende realizar:

1. A parte del diseño de la aplicación en sí, entender y aprender los diversos conocimientos sobre las tecnologías que se usan en la actualidad para crear servicios y aplicaciones web.
2. Aprender a manejar distintos tipos de framework que facilitan y agilizan las cargas de trabajo.
3. Analizar las técnicas de recomendación turística y de planificación, revisando su adecuación y / o aplicabilidad al problema.
4. Entender el funcionamiento interno de los servicios web, tanto a nivel de petición como de tratamiento.

Con todos estos objetivos específicos se pretende conseguir:

1. Obtener los gustos y preferencias de los usuarios sobre las actividades de un viaje.
2. Planificar un viaje para un grupo de usuarios que detalle las actividades que se van a hacer cada día del viaje.
3. Verificar que los usuarios quedan satisfechos con la planificación recomendada por la aplicación.
4. Realizar un proceso de retroalimentación basado en las valoraciones de los usuarios para adaptar al gusto general del grupo.

Llegando de esta manera, a construir un sistema que proporciona servicios web que resulten lo más intuitivo posible para elaborar planes turísticos sobre una comunidad autónoma de España, en este caso concretamente, fue elegida Andalucía.

1.2. Estructura de la memoria

El contenido de esta memoria está organizado de la siguiente forma:

- En el segundo y tercer apartado se describirá brevemente el estado del arte de los sistemas de recomendación para viajes en grupos, y descripción de las tecnologías aplicadas al proyecto y de las metodologías usadas en él.

- En el cuarto apartado, se describirá la aplicación desde el punto de vista del usuario, exponiendo un ejemplo de uso, y posteriormente una descripción funcional.

- En el quinto apartado se describirá a fondo la arquitectura del sistema, así como los algoritmos empleados en el sistema.

- A continuación, se expondrá una evaluación con usuarios realizada sobre la interfaz web creada.

- Como parte final se expondrán las conclusiones y visión futura del trabajo realizado, así como las aportaciones de cada estudiante.

- Finalmente, se encontrará una serie de apéndices como apoyo a diversos apartados comentados con anterioridad, y la bibliografía consultada.

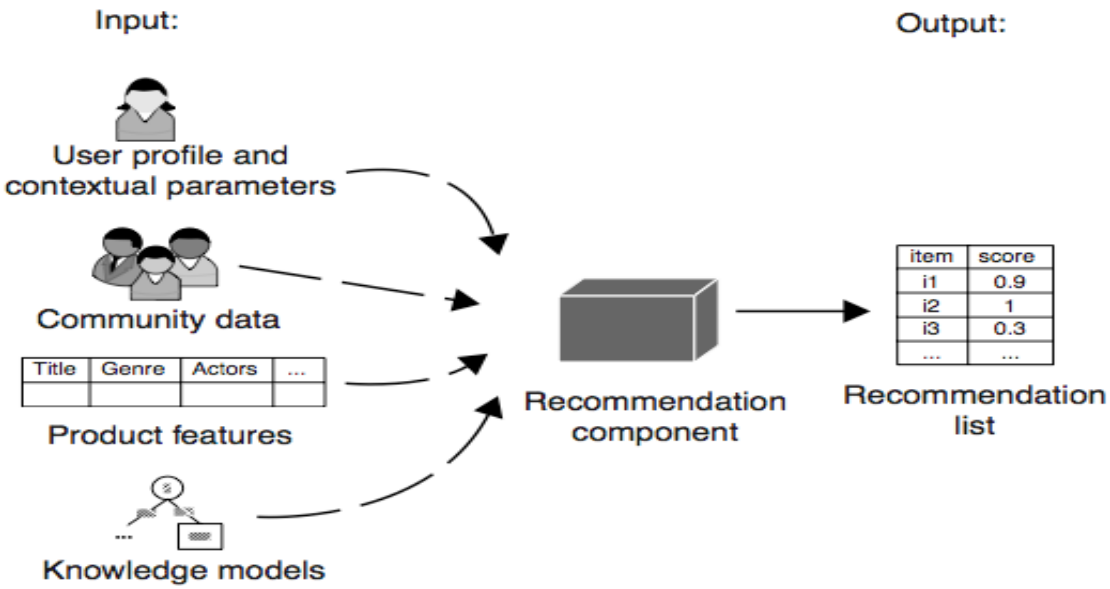
2. Estado del Arte

En este capítulo se expondrá una introducción a los sistemas de recomendación existentes, y su clasificación, así como ejemplos de este tipo de sistemas que están relacionados con el proyecto.

2.1. Sistemas de Recomendación

A lo largo de nuestra vida, nos encontramos frente a diversas situaciones en las cuales se tiene que tomar una o varias decisiones respecto a diversas alternativas, como, por ejemplo, ir a Sevilla o Córdoba, con amigos o en familia, llegando muchas veces a ser una tarea tediosa ya que pueden surgir muchas alternativas y/o propuestas. A consecuencia de ello surgen los sistemas de recomendación, cuyo principal objetivo es el de filtrar información que cumpla los requisitos del usuario, facilitando así la toma de decisiones.

Actualmente, estos sistemas son muy frecuentes, llegando a encontrar sistemas de recomendación de todo tipo (viajes, libros, películas, restaurantes, etc.), y cada vez están más en auge, sobre todo los sistemas de recomendación relacionados con el ocio, tanto basadas en un único individuo, como en varios.



2.1 SISTEMAS DE RECOMENDACIÓN

Los sistemas de recomendación se pueden clasificar en:

2.1.1. Sistemas de recomendación individuales

Sistemas que se centran en la recomendación para un único usuario en base a sus preferencias. Estos sistemas pueden clasificarse en:

- ***Recomendación colaborativa***

  Estos sistemas emplean una técnica de filtrado colaborativo, en el cual se emplean múltiples agentes, fuentes de datos, etc.; evaluando y seleccionando productos a partir de las opiniones de otros usuarios [1]. Esta técnica, aunque lleva poco implementándose, puede decirse que es la predecesora del concepto de compartición de opiniones.

Un ejemplo claro sería el proceso que llevan a cabo las personas antes de decantarse por comprar algo, que muchas veces implica una breve "investigación", la cual nos lleva a preguntar a nuestro entorno sus opiniones respecto a lo que se desea comprar, y previamente en base a esas opiniones se filtran las que más conviene.

Un buen sistema de recomendación colaborativo, es aquel que nos proporciona recomendaciones de calidad, para ello es necesario un disponer de un buen algoritmo de filtrado colaborativo. Estos se pueden clasificar en:

- Basados en memoria

  Este tipo de algoritmo emplea la base de datos completa para generar una predicción óptima, ayudándose de métodos estadísticos para determinar usuarios con un historial de valoraciones similar al usuario actual (vecinos) y a continuación, mediante un conjunto de algoritmos combinan las preferencias para realizar las recomendaciones. [2]

- Basados en modelo

  Este tipo de algoritmos desarrollan un modelo de puntuaciones de los usuarios sobre los productos, diferenciándose del anterior en que no se emplean métodos estadísticos, sino probabilísticos de aproximación (predicción), para ello se calcula un valor esperando en base a una predicción del usuario para cada producto en función de la clasificación obtenida a priori. [3]

- ***Recomendación basada en contenido***

Los sistemas de recomendación basados en contenido son aquellos que realizan la recomendación en base a la descripción de los productos a recomendar, comparándolos con las preferencias del usuario. El razonamiento que emplean estos sistemas se basa en que, si a un usuario le gustan los productos de un determinado dominio, también le gustarán otros productos de características similares. [1].

- ***Recomendación basada en casos: CBR***

Este tipo de sistemas tienen relación con el anterior, ya que en el proceso de solucionar nuevas incógnitas se basa en incógnitas anteriores. Estos cuentan con una base de casos de problemas resueltos con anterioridad, resolviendo los problemas nuevos adaptando los ya existentes. [4]

Este método se basa principalmente en:

1. Recuperar los casos con mayor similitud.
2. Reutilizar el caso seleccionado para ser empleado.
3. Revisar la solución, y adaptarla a lo que se desea.
4. Recordar la solución, y almacenarla en la base da casos.

- ***Recomendación basada en conocimiento***

Mediante estos sistemas se calculan de forma independiente las calificaciones de los usuarios: ya sea en forma de similitudes entre las necesidades del cliente y los artículos o en función de reglas de recomendación explicitas. [5]

Estos sistemas, no sólo filtran resultados, sino que de forma personalizada ayudan a los usuarios a alcanzar sus objetivos, los cuales se encuentran dentro de múltiples opciones.

Estos sistemas pueden clasificarse en:

- Basado en restricciones.
- Basados en casos.

- ***Recomendación basada en demografía***

Estos tipos de sistemas clasifican las preferencias de usuarios en base a sus características demográficas, y basándose en ellas posteriormente. [6]
Sin embargo, este tipo de sistema es a veces difícil de implementar, ya que la recogida de datos puede ser compleja, ya que no todos los usuarios son propensos a facilitar información personal.

- ***Recomendación basada en utilidad***

No son propensos a construir generalidades a largo plazo, sino más bien compara la necesidad del usuario con el conjunto de opciones disponibles, mediante una función de utilidad de cada objeto para el usuario, tal función será su perfil, y posteriormente se emplean técnicas de satisfacción de restricciones para escoger la mejor opción. [7]

- ***Recomendación híbrida***

  Es aquel sistema que combina múltiples técnicas en una sola consiguiendo una participación activa de todas ellas. Dentro de este sistema se definen hasta siete métodos de combinación de técnicas. [7]

  - Método de Conmutación

    En vez de ejecutar todas las estrategias simultáneamente, el sistema emplea algún criterio para la conmutación entre ellas, y cuando se cumplen ciertas condiciones el sistema emplea una estrategia y/o recurre a las restantes.

  - Método de Ponderado

    Se combinan los resultados de las distintas técnicas de recomendación que componen el sistema, y se obtiene una puntuación para cada producto en base a la puntuación asignada a cada uno de ellos por las distintas estrategias. El producto con mayor puntuación será el que se ofrezca al usuario.

2.1.2. Sistemas de recomendación grupales

Los sistemas de recomendación individuales sugieren una serie de productos a un usuario. Sin embargo, se quedan obsoletos a la hora de recomendar un mismo producto a un grupo de usuarios.

Por ello, surgen los sistemas de recomendación para grupos, que tratan de recomendar a un grupo de personas sobre una serie de productos a partir de las preferencias individuales de cada uno de ellos, ayudando al grupo en el proceso de toma de decisiones. [8]

Sin embargo, este tipo se sistemas no son tan sencillos con los de recomendación individual, ya que ha de ser un sistema dinámico y diverso, que se adapte a la mayoría, ya que un grupo puede estar formado por

diversas personas con distintos intereses. Por eso, no sólo ha de captar las preferencias de cada uno, sino también los métodos de decisión llevados a cabo.

Existen diversas soluciones que permiten dar diferentes resultados en base a las combinaciones de preferencias que se utilicen.

- Mezcla de recomendaciones individuales

  Método simple de agregación donde a partir de un conjunto de recomendaciones individuales de cada miembro del grupo, se obtiene un único listado. [9]

- Crear agregaciones

  Dar valoraciones y/puntuaciones por cada miembro del grupo, es decir, para cada miembro, el sistema escoge el conjunto de recomendaciones que tenga la mayor puntuación. Mediante una fórmula escogida se agregan todas las recomendaciones del conjunto final y se determina cuál es la mejor.

- Tratar de forma diferente a cada miembro del grupo.

  Para ello se ha de prestar atención a aquellos usuarios que tienen características diferentes al resto, asignando distintos pesos a los miembros en función de la influencia, dando más importancia a algunos de ellos, ya que puede ser más determinante que los demás.

- Crear un modelo del grupo

  Para ello se trata al grupo como si fuese un único usuario, de tal forma que se unen las preferencias de cada miembro del grupo, asemejándose luego a los sistemas de recomendación individuales. [10]

  Para construir el modelo, existen diversos métodos, entre ellos:

    - En base al comportamiento global de los miembros.

- En base a la agregación de modelos individuales de cada miembro
- En base a la división del grupo en subgrupos.

En relación con este proyecto, una buena forma de llevar a cabo la recomendación grupal, es la creación de agregaciones, en este caso a cada actividad y/o viaje, a continuación, se explicará más a fondo los objetivos necesarios para crear las agregaciones:

- Maximizar satisfacción media

  Se intenta que el grupo quede satisfecho en la medida de los posible, donde cada miembro del grupo tiene la misma importancia que el resto.

$$R_i = media(r_{ij}) = \frac{1}{n}\sum_{j=1}^{n} r_{ij}$$

- Minimizar miserias

  Se intenta que ningún miembro del grupo quede insatisfecho, por eso, se emplea cuando algún miembro del grupo está insatisfecho.

$$R_i = MAX(min_j r_{ij})$$

- Maximizar satisfacción

  Se intenta que el usuario más satisfecho con una actividad tenga más prioridad que los demás.

$$R_i = MAX(max_j r_{ij})$$

Como se ha comentado, estos sistemas son más complejos, que los individuales, y por ese motivo, se han de desarrollar técnicas para obtener información, similares a los de los sistemas individuales, entre ellas destacan:

- Preferencias especificadas explícitamente

  Los usuarios proporcionan preferencias al sistema, que serán utilizadas con posterioridad. Un ejemplo claro sería cuando un usuario da la valoración de un determinado producto.

- Preferencias no especificadas explícitamente

  Los usuarios no proporcionan preferencias al sistema, si no que estas son recolectadas por hábitos de navegación. El número de visitas a un determinado producto, sería un ejemplo de tal caso.

Sin embargo, la información recabada a veces no es suficiente, y se tiende a combinar preferencias añadiendo también, métodos de agregación, como los explicados con anterioridad, para poder satisfacer a un mayor número de usuarios.

2.2. Sistemas de recomendación relacionados con el ocio y turismo

El sector turístico es uno de los más importantes, el cual va evolucionando con el paso del tiempo, España, por ejemplo, es líder en dicho sector, así lo acreditan las cifras de turistas internacionales, de ingresos por turismo y la aportación de la actividad turística a la economía de nuestro país. [11]

La evolución de este sector viene precedida por los avances en tecnología, innovación y digitalización, tanto en nuestra vida privada, pública y profesional. Es probable que, la cuarta revolución industrial esté cerca, y los medios de comunicación, entretenimiento e información son quizás las responsables de ello; proporcionan herramientas digitales, servicios, aplicaciones y contenidos, que nos son facilitados en cualquier momento, y en cualquier lugar.

Tales avances, son causantes de la era digital, dónde los Smartphone, Tablets y otros dispositivos hacen que las relaciones con elementos de la vida cotidiana cambien, por ejemplo, a la hora de planear un viaje. Antes se dependía de una agencia de viajes que ayudase conseguir el mejor destino con las mejores actividades, sin embargo, actualmente las personas, en todo el mundo, emplean su tiempo, no en ir a la agencia, si no explorar lo que internet ofrece. Como consecuencia, en la última década han surgido

diferentes aplicaciones que ayudan a tal labor, y éstas se valen de sistemas de recomendación, los cuales cuentan con un gran volumen de información turística, donde cualquier usuario puede encontrar información detallada de cualquier destino, y las actividades que puede realizar (horarios, precios, teléfonos, etc). [12]

Cabe destacar, que no todos los sistemas, por mucha cantidad de información que ofrezcan, son válidos para todos los usuarios, ya que tanta sobreinformación hace que se necesite mucho tiempo y esfuerzo.

Ejemplos de este tipo de sistema, pueden ser la Guía Repsol, TripAdvisor, Mi Nube, Fever, etc. Estas aplicaciones muestran formularios de búsqueda que dan como resultado listados de lugares de interés que hay en un lugar determinado, ordenados por la popularidad, proporcionando la información completa sobre la actividad e incluso en algunas páginas con la opción de organizar rutas seleccionando manualmente los sitios por el usuario. Ofrecen también reservas complementarias al viaje, como, por ejemplo, reservas de hotel, de restaurantes y de transporte para llegar al destino.

Pero toda la información que aparece es de forma individual y sin coherencia, de forma que a un usuario que pretende planear un viaje, toda esa información le dificulta a la hora de selección de las actividades más atractivas, complicando el proceso de planificación de la ruta, y llevándole incluso a sentirse abrumado con tanta información.

2.2.1. Guía Repsol

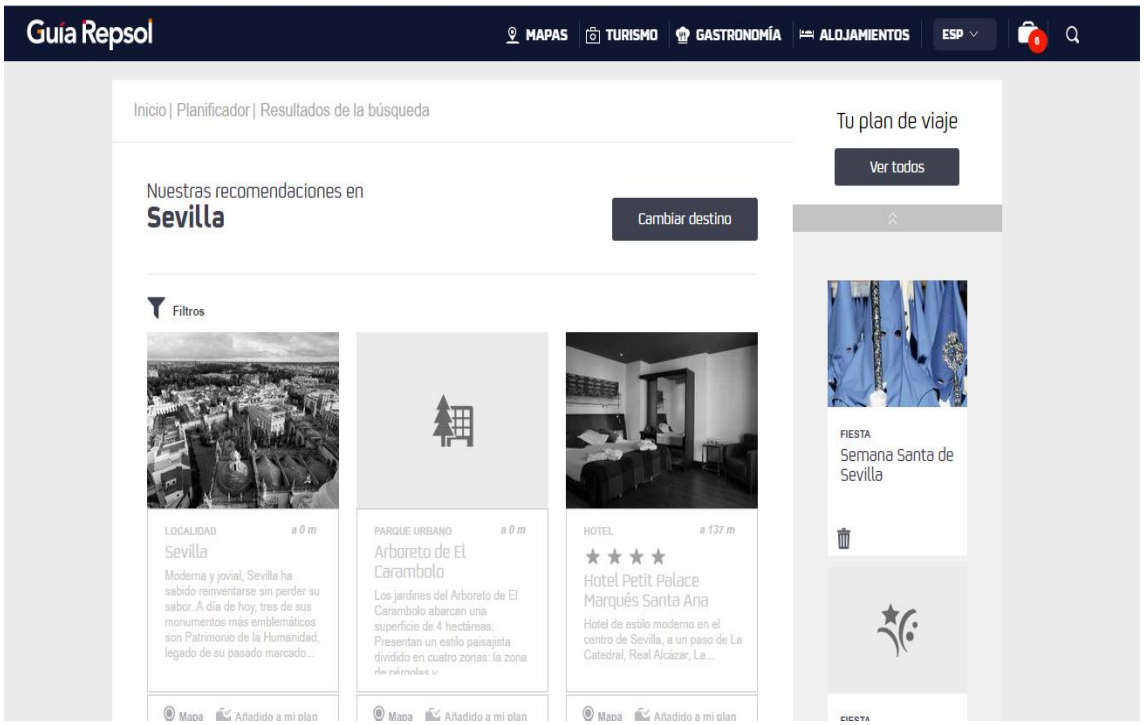
Uno de los ejemplos tomado como referencia es el planificador de viaje de Guía Repsol[1], una página web que aporta una planificación de viaje al usuario,

Tal como se indica en la figura 2.2, esta página web proporciona los lugares populares (filtro colaborativo) de una determinada provincia

[1] http://www.guiarepsol.com/es/

elegidos por el usuario. Ordenado por la localización más cercana al punto céntrico de la provincia.

También, permite al usuario arrastrar las actividades / lugares a su plan, con el fin de tener la lista de lugares a visitar de la provincia, ordenados según el tiempo de añadido al planificador. Cabe destacar que dispone de un filtro para buscar lugares según la categoría, facilitando al usuario la localización de actividades. Una vez seleccionada las actividades, el planificador ofrece también la posibilidad de visualizar en el mapa, la localización de cada uno de los lugares añadidos al plan de la provincia.



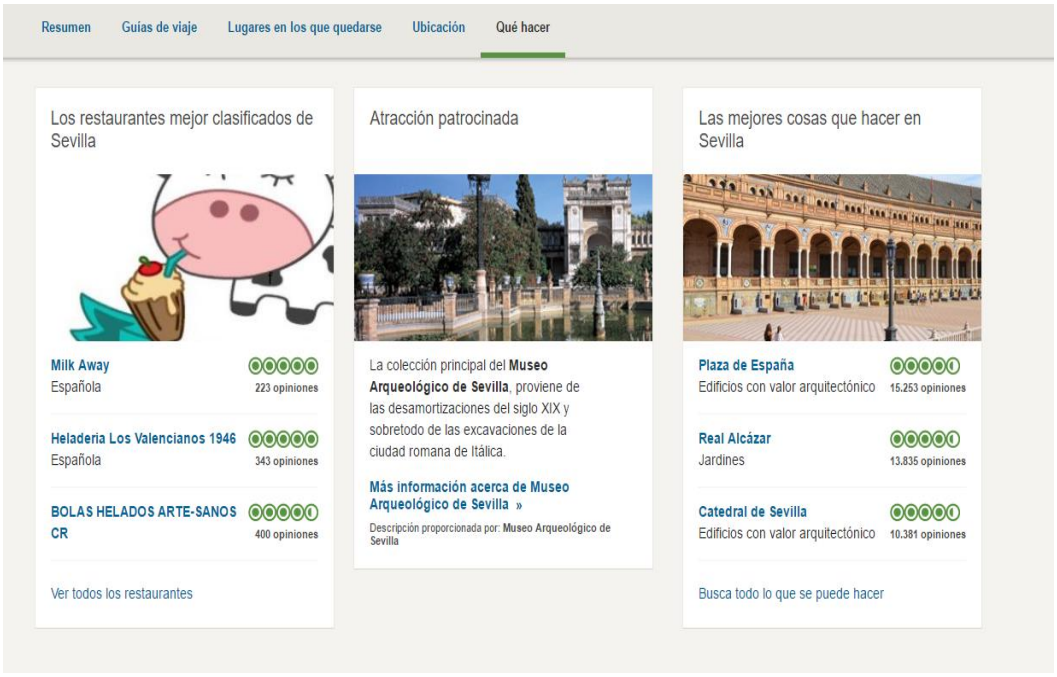
2.2 GUÍA REPSOL

Sin embargo, esta aplicación no es capaz de elaborar una ruta con las actividades escogidas, ni tampoco permite la colaboración en grupo, sólo muestra información de las actividades.

2.2.2. TripAdvisor

Otro ejemplo relevante es el sistema de recomendación de actividades de TripAdvisor[2]. Como se puede ver en la figura 2.3, esta página web no solo proporciona posibilidad de buscar actividades clasificadas por tipo en una única provincia, sino que además sugiere las actividades y los lugares más valorados por los usuarios registrados, utilizando técnicas de filtrado colaborativo, el cual consiste en hacer recomendaciones basados según las valoraciones numéricas y descriptivas sobre los sitios de las personas registradas en la aplicación. Esta información se usa para asegurar que las actividades sugeridas en la página principal, son de alta valoración por mayor parte de los usuarios visitados.

Pero, esta aplicación no es capaz de recomendar actividades de diversos destinos, ni tampoco crear una ruta idónea para las actividades que recomienda, y se centra sólo en un único usuario.



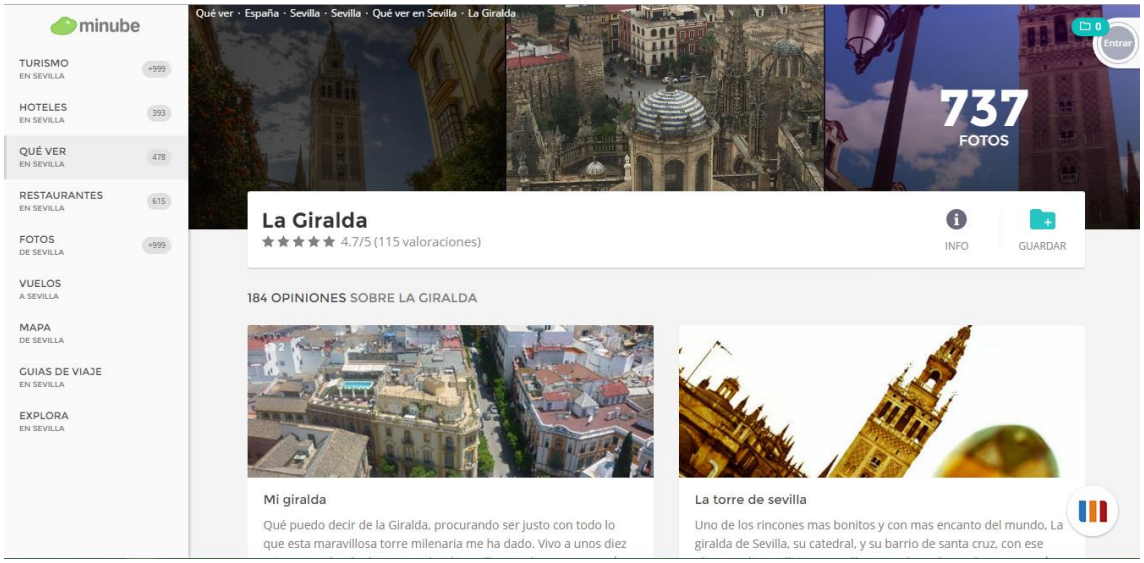
2.3 TRIPADVISOR

[2] https://www.tripadvisor.es/

2.2.3. Mi Nube

En esta aplicación, Mi Nube[3], al igual que las anteriores los usuarios disponen de un amplio catálogo de actividades filtrados mediante filtros colaborativos, y, además, los usuarios reciben recomendaciones de lugares a visitar en base a sus intereses personales. El sistema de recomendaciones que utiliza analiza la actividad del usuario en la aplicación, en este caso se fija en los rincones que el usuario ha guardado como favoritos para en base a sus características poder ofrecer alternativas relacionadas en base a los nuevos destinos que esté interesado en visitar. El resultado es que se facilita al usuario descubrir lo mejor del contenido de la web.

Sin embargo, no recomienda rutas, ni personalizar las actividades y/o resultados al gusto del usuario, y mucho menos recomendaciones grupales.



2.4 MI NUBE

[3] http://www.minube.com/

2.2.4. Fever

Fever [4] es una aplicación móvil, que recomienda planes de ocio y entretenimiento, en base a unos hashtags (preferencias) proporcionados por el usuario, para ello, al igual que en las aplicaciones anteriores se vale de filtros colaborativos, además filtra actividades patrocinadas por empresas. También permite la reserva y/o compra directa a través de la aplicación. Aunque cabe tener en cuenta, que no es capaz de recomendar rutas, ni es un sistema grupal.

2.5 FEVER

Sin embargo, las aplicaciones anteriores, filtran y recogen actividades acordes con las preferencias de un usuario, en otras palabras, sólo sugieren actividades en base a sus preferencias. Aportan información muy general, muchas veces sin valor para el usuario, es decir, que ofrece sistemáticamente la misma información a todos los usuarios independientemente del perfil de estos, y muestran grandes volúmenes de información que el usuario debe procesar posteriormente para seleccionar aquellos elementos que pueden ser de su interés.

[4] http://www.feverup.com/?locale=es

La aplicación OléTrip se centrará en facilitar el proceso de planificación del viaje aportando recomendaciones según las preferencias obtenidas de los usuarios. Ahorrando así, el trabajo de organización manual del viaje por parte de los usuarios, y solucionando así, la escasez de sistemas de recomendación grupales, y llegando también a recomendar rutas óptimas, y dinámicas, que se modifiquen según las preferencias de usuarios.

Será un sistema de recomendación de rutas turísticas planificadas en la Comunidad Autónoma de Andalucía. Que empleará filtros colaborativos, filtros basados en contenido, agregaciones, etc.; e intentará que, con la ayuda de un usuario creador, todos los miembros del plan queden satisfechos.

3. Tecnologías aplicadas

En este punto se describirán las tecnologías usadas para el desarrollo de nuestra aplicación, *Olétrip* y detallándose los distintos mecanismos y algoritmos empleados para obtener los resultados requeridos.

Para el proyecto se ha propuesto elaborar tanto la parte del servidor como un prototipo de cliente para mostrar el funcionamiento de los servicios, así como un posible uso de los mismos, basándose en tecnologías web.

3.1 TECNOLOGIAS USADAS

3.1. Backend – JAVA EE

Java EE (Java Enterprise Edition) es un conjunto de especificaciones de APIs (Application Programming Interface) Java que nos permite diseñar aplicaciones distribuidas de tipo multicapa sobre Web.[14]

En este caso, se ha optado por implementar servicios RESTful sobre el protocolo HTTP, recibiendo peticiones mediante direcciones concretas y retornando una respuesta en formato JSON; más adelante se describirá cada uno de estos conceptos.

Para gestionar el ciclo de vida del proyecto Java EE se ha usado la herramienta de Maven que aporta ventajas como:

- Estandarización de las estructuras de los directorios de trabajo.
- Estandarización del ciclo de vida del proyecto.
- Reutilización como de plug-ins o herencia de configuración.
- Gestión de dependencias.

Para el desarrollo se ha utilizado:

- Eclipse Mars versión 2.
- JRE 1.8.0_73
- Maven versión 3.3.3

3.1.1. Jersey

Para construir los servicios web de la aplicación se ha empleado Jersey, que es una implementación y mejora de la API JAX-RS (Java API for RESTful Web Services [15][20]) para diseñar aplicaciones Web con métodos definidos mediante uso explícito de los métodos HTTP basado en el protocolo definido por RFC2616. Los protocolos que se emplearán son los siguientes:

- POST para la creación de los recursos, así como la actualización de los mismos.
- GET para la obtención de los recursos.

Jersey ha sido elegido por ser más simple y fácil de usar frente a los servicios SOAP y los servicios web tipo WSDL. Existen diversas grandes empresas que están migrando a esta tecnología orientada a dichos recursos.

Un servicio RESTful hace referencia a un servicio web que implementa la arquitectura REST (Representational State Transfer), el cual cumple los siguientes diseños fundamentales:

- Arquitectura cliente – servidor: Separan los dos agentes básicos para el intercambio de información, estos deben ser independiente entre sí.

- Protocolo sin estado: Se define como, protocolo de comunicación que trata cada petición como una transacción independiente de cualquier comunicación anterior.
- Interfaz uniforme: Permite al servidor conservar su independencia de tratamientos de la información frente al cliente.
- Sintaxis universal: Cada recurso es únicamente tratables mediante su URI (identificador de recursos uniformes) como si fuera directorios por lo que deben ser muy intuitivas.
- Mensajes autodescriptivos: Las peticiones al servidor deben retornar una respuesta directa a la operación requerida.

3.1.2. Hibernate ORM

Para la constante necesidad de persistir, borrar y modificar datos hemos recurrido a Hibernate ORM (Object-Relational mapping)[16] que es un framework de código libre bajo licencias para entornos Java desarrollada por Gavin King en 2001.

Hibernate se ocupa del mapeo entre las clases de entidades en Java y las tablas de la base de datos, incluyendo el mapeo entre los tipos de datos de Java y tipos de datos de SQL. A parte de lo comentado, se encarga también de facilitar la consulta y recuperación de datos, frente a métodos tradicionales, esto reduce significadamente el tiempo para los manejos de datos.

Para el mapeo de las entidades hemos recurrido a anotaciones de JPA (Java Persistence API), el cual también implementa Hibernate. La JPA nos proporciona soluciones sobre las discordancias entre los modelos relaciones y objeto Java, como los siguientes:

- Granularidad: Los modelos de los objetos Java suele tener mayor nivel del detalle que el modelo relacional de base de datos.

- Herencia: estos no son compatibles con todos los tipos de bases de datos relacionales, los que son compatibles también necesitan ser simulados con complejos sistemas.
- Asociaciones: los modelos relaciones no puede determinar relaciones múltiples basado en un objeto modelo.
- Navegación de datos: La navegación entre objetos (normalmente jerárquicos) distintos difiere a la navegación tradicional en tablas de la base de datos.

El manejo de datos Hibernate proporciona métodos predefinidos que facilitan las operaciones más frecuentes sobre los datos (Criteria). Estos pueden ser, por ejemplo: save, delete, update, merge, saveOrUpdate, etc… Pero para un manejo más concreto de datos o consultas estáticas se ha de recurrir a HQL (Hibernate Query Language) que es similar a sentencias SQL, pero con alguna diferencia:

- Hace referencia al objeto y no a una tabla concreto, por lo que el resultado de la consulta podría ser una clase objeto.
- Es sensible a las mayúsculas y minúsculas.

3.1.3. Log4j

Se ha empleado el framework Log4j (Log For Java)[17] para centralizar todos los mensajes de información de nuestra aplicación en archivos logs. Estos ficheros almacenan todas las trazas o errores a lo largo del funcionamiento de los distintos servicios.

Este framework permite configurar las trazas de salidas en un formato concentro y por distintos niveles de prioridad:

- FATAL: se utilizan para mensajes críticos de la aplicación.
- ERROR: se utilizan en los mensajes de errores producidos por ejemplo por algún parámetro de entrada o inconsistencia de algunos datos de entrada.

- WARN: se emplea para avisar de alguna alerta sobre determinado eventos o funcionamientos que no afecta el funcionamiento normal de la aplicación.
- INFO: se utilizan para mostrar mensajes informativos.
- DEGUB: sirve para mostrar los mensajes de depuración en las fases de pruebas o pre-producción.
- TRACE: su funcionamiento es similar que DEBUG, pero con un mayor nivel de detalle.

Una vez desplegada la aplicación en el contenedor redirige todas las trazas o informaciones necesarias para registrar tanto el uso como los parámetros de entrada, estados de ejecución, etc., al log para algún evaluación o depuración a posteriori.

```
2016-05-19 21:41:34 INFO  SiteServices:37 - CALLING GET_SITES FUNCTION (GET)
2016-05-19 21:41:34 INFO  SiteServices:37 - CALLING GET_SITES FUNCTION (GET)
2016-05-19 21:42:22 INFO  TripServices:257 - CALLING CALCULATE_TRIP_NEW2 FUNCTION (POST)
2016-05-19 21:42:22 INFO  TripServices:258 - ************************************************
2016-05-19 21:42:22 INFO  TripServices:259 - *Sites elegidos:[4844, 4915, 7809, 6744, 6695, 7898, 7928]
2016-05-19 21:42:22 INFO  TripServices:262 - *Trip days:2
2016-05-19 21:42:22 INFO  TripServices:263 - *Trip transport:driving
2016-05-19 21:42:22 INFO  TripServices:264 - *Trip startTime:32400
2016-05-19 21:42:22 INFO  TripServices:265 - *Trip endTime:72000
2016-05-19 21:42:22 INFO  TripServices:266 - *Trip Cities:[cadiz, huelva, sevilla]
2016-05-19 21:42:23 INFO  TripServices:275 - *Resultado:{"day_2":[{"activity_3":{"duration":900000,"distance":827
2016-05-19 21:42:23 INFO  TripServices:276 - ************************************************
2016-05-19 21:42:43 INFO  TripServices:293 - CALLING SAVE FUNCTION (POST)
2016-05-19 21:42:44 INFO  TripServices:335 - CALLING IS_OWNER FUNCTION (GET)
2016-05-19 21:42:44 INFO  TripServices:372 - CALLING VERIFY_TOKENS FUNCTION (POST)
2016-05-19 21:42:44 INFO  TripServices:62 - CALLING GET_TRIP_INFO FUNCTION (GET)
2016-05-19 21:42:44 INFO  ActivityServices:208 - CALLING GET_USER_VOTE FUNCTION (GET)
```

3.2 TRAZA LOG4J

3.1.4. Server – Tomcat y MySQL Server

Olétrip se encuentra alojado en un servidor de la Facultad de Informática de la Universidad Complutense, proporcionado por el personal de laboratorio para investigaciones y temas relacionados con el estudio. La configuración de éste, se encuentra en el **Apéndice D.**

El servidor que se ha empleado es el Apache Tomcat [18], el cual es un contenedor de servlets desarrollado por Apache Software Fundation. En concreto la versión 8 que implementa las especificaciones de Servlet 3.0 y JSP 2.2.

La primera ventaja de todas es que es gratuito y los usuarios disponen de libre acceso al código fuente bajo la licencia de Apache Software License.

Además, es bastante fácil de configurar y dispone de abundante información.

El sistema de gestor de base de datos instalado fue MySQL server versión distribuida 5.5.47 [19]. Este DBMS ha sido elegido por su facilidad de configuración y utilización. El usuario para el acceso y modificación de la base de datos "oletrip_tfg" ha de coincidir con la configuración de Hibernate. Hay que crear una base de datos vacía a priori para el correcto funcionamiento de la aplicación.

3.1.5. Otros

Otros framework que se han empleado para el desarrollo fueron:

- MySQL Connector: Biblioteca principal que se encarga de conectar la aplicación con la base de datos.
- Javax Mail: Este framework permite mandar mensajes de invitación a los distintos usuarios invitados.

3.2. Frontend – JavaScript

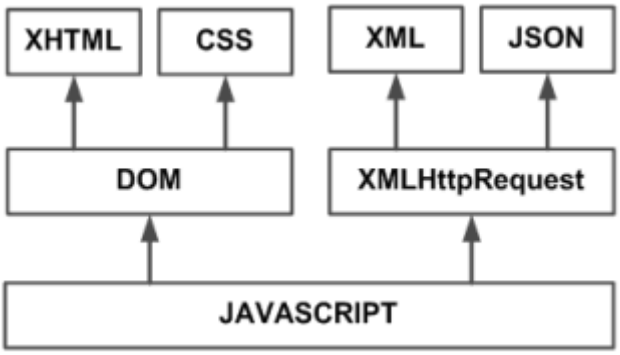
3.2.1. JavaScript, AJAX y Jquery /Jquery IU

Lenguaje de programación interpretado que Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

AJAX [21] es una tecnología de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad

de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

AJAX es una tecnología asíncrona, en el sentido de que los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página, aunque existe la posibilidad de configurar las peticiones como síncronas de tal forma que la interactividad de la página se detiene hasta la espera de la respuesta por parte del servidor.



3.3 TECNOLOGÍAS QUE FORMAN AJAX

jQuery [5] es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

jQuery UI es una biblioteca de componentes de jQuery que le añaden un conjunto de plug-ins, widgets y efectos visuales para la creación de aplicaciones web, en nuestro proyecto se han empleado:

- Sortable [27]: Ordena una lista de elementos.
- Draggable [28]: Hace al elemento arrastrable.
- Droppable [29]: Permite que el elemento responda a elementos arrastrables.

[5] https://jquery.com/

- DatePicker [30]: Calendarios personalizados

3.2.2. JSON

JSON (JavaScript Object Notation)[22] es un sistema de intercambio de datos ligero que facilita a las personas la comprensión de la información, fácil de parsear y generar para las máquinas. El cual devuelve un formato de texto comprensible para cualquier tipo de lenguaje, por eso, JSON es ideal para el intercambio de información entre lenguajes.

JSON permite la representación de números, cadenas, booleanos, null, vectores y objetos, representados por una colección de pares clave/valor. Por tales características, se ha decidido emplear para la comunicación entre la interfaz web y la API de servicios de nuestro sistema.

```
{
      createdTrip: "[1, 2, 3]",
      listTrip: "[7,8]",
      name: "User",
      bornDate: "1993-12-28",
      email: "user@email.com"
}
```

3.2.3. HTML5 y CSS3

HTML (HyperText MarkUp Lenguage) [23], es un lenguaje de marcado para la elaboración de páginas web, un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros.

Este es un lenguaje sencillo, el cual utiliza "etiquetas", rodeadas por corchetes angulares (<, >, /), el cual puede describir hasta cierto punto la apariencia del documento, el cual puede incluir o hacer referencia a un tipo de programa llamado script (JavaScript).

En este proyecto, se empleará la quinta versión, que ofrece elementos y atributos, que reflejan el uso típico de los sitios web modernos. Algunos de ellos son técnicamente similares a las etiquetas y , pero tienen un significado semántico, como por ejemplo <nav> (bloque de navegación del sitio web) y <footer>.

Además, a este lenguaje se le puede agregar estilo, el cual puede estar en el propio código, o en un fichero auxiliar, el cual se denomina hoja de estilo en cascada o CSS, además se puede definir cuestiones relativas a colores, fuentes, márgenes, tamaños, imágenes de fondo, posicionamientos, etc.

3.2.4. Bootstrap

Framework o conjunto de herramientas de Código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.

Bootstrap [24] es modular y consiste esencialmente en una serie de hojas de estilo LESS que implementan la variedad de componentes de la herramienta. Una hoja de estilo llamada bootstrap.less incluye los componentes de las hojas de estilo. Los desarrolladores pueden adaptar el mismo archivo de Bootstrap, seleccionando los componentes que deseen usar en su proyecto.

Además, tiene a disposición un sistema de grid, con cuatro variaciones para hacer uso de distintas resoluciones y tipos de dispositivos: teléfonos móviles, formato de retrato y paisaje, tabletas y computadoras con baja y alta resolución (pantalla ancha). Esto ajusta el ancho de las columnas automáticamente, y permite así que el diseño web sea adaptable (Responsive Design), es decir, un diseño que se adapta a cualquier tipo de dispositivos.

3.2.5. Google APIs

Para poder mostrar mapas, rutas, y cálculos de tiempos, se han empleado diferentes API proporcionadas por Google, entre ellas Google Maps JavaScript API, y Google Maps Distance Matrix API, las cuales permiten construir mapas personalizados usando mapas con estilos, edificios en 3D, planos para pisos de interiores, indicaciones de varios modos y más. Simplemente, se necesita estar registrado en Google Developers, y obtener tu clave de producto. [25]

3.2.6. Librerías de formato

Cabe destacar, que además se han empleado las siguientes librerías:

- ***Slick.js***

Carousel adaptativo empleando jQuery que soporta múltiples puntos de interrupción, CSS·, transiciones, eventos, etc. El cual se ha empleado para mostrar los diferentes resultados por día de la ruta recomendada.[26]

- ***FontAwesome***

Proporciona múltiples iconos vectoriales escalables, que puede ser costumizados fácilmente. Simplemente añadiendo en el código HTML la etiqueta `<i>` con la clase fa: [31]

```
<i class="fa fa-open" aria-hidden="true"></i>
```

- ***Moment.js***

Librería de parsing, validación, manipulación y formato de fechas, que nos permite mostrar las fechas en nuestra web de la siguiente manera: 6 JUNIO 2016, simplemente pasándole la fecha en milisegundos.[32]

```
moment(1465171200000). locale('ES').format('DD MMMM YYYY');
```

4. Descripción funcional

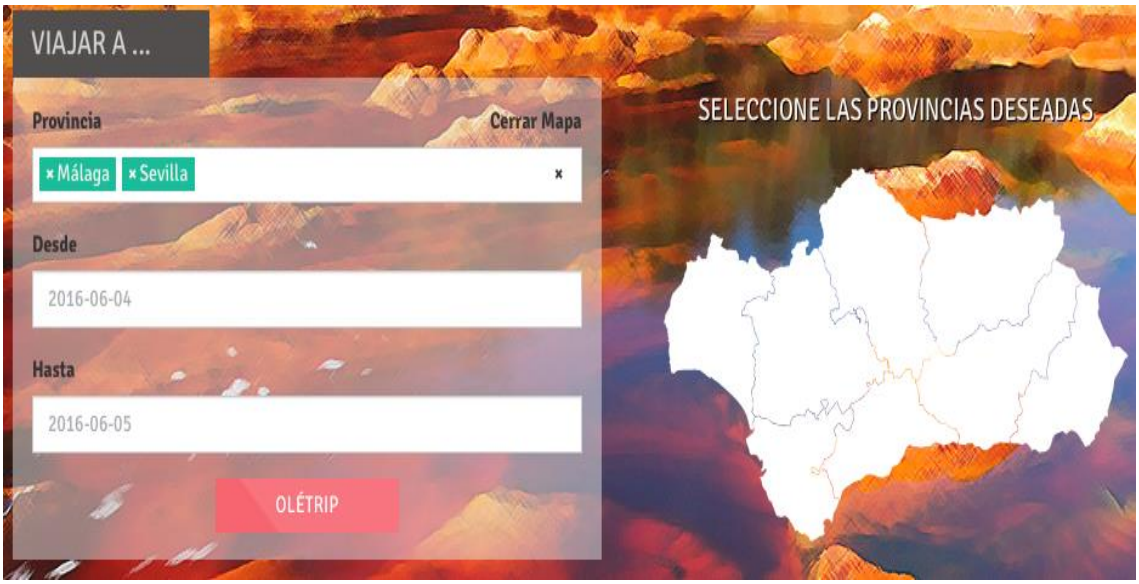
En el este capítulo, se expondrán un posible ejemplo de uso, es decir, como poder usar la aplicación Olétrip en base a una situación. Además de los posibles casos de uso de la aplicación.

4.1 SISTEMA RECOMENDADOR

4.1. Ejemplo de uso

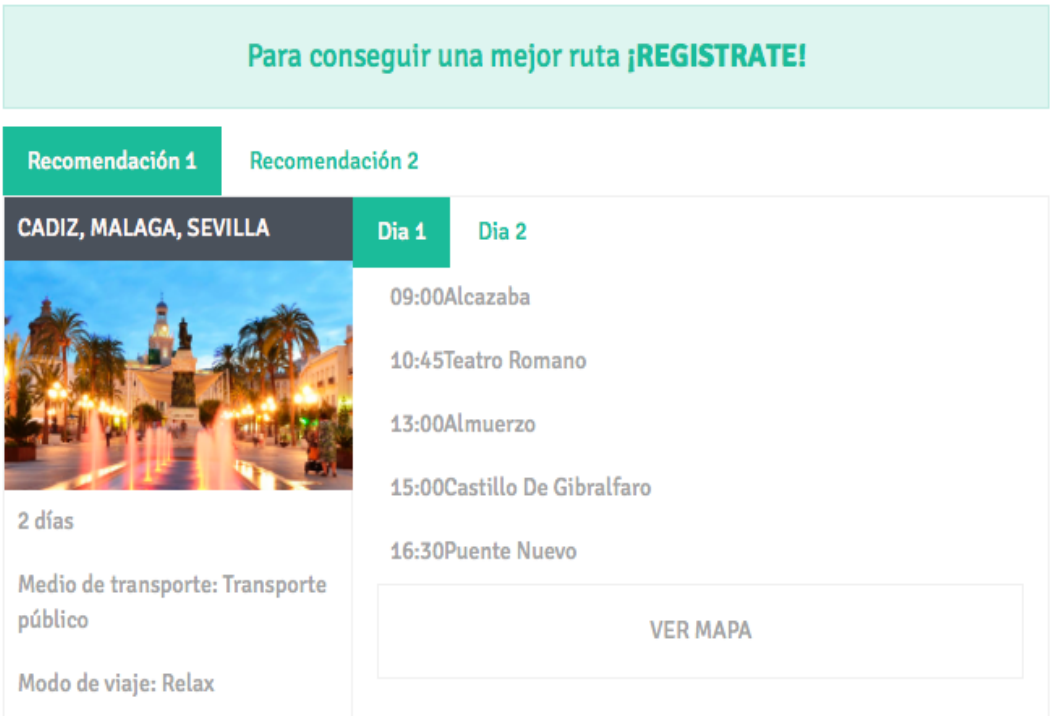
Ana quiere organizar un viaje para irse de vacaciones con un grupo de amigos un fin de semana, pero no tiene claro dónde ir exactamente, para ello, decide entrar en Olétrip para buscar ideas.

Al entrar, ve un cuadro que pide seleccionar las provincias a visitar y la duración del viaje. Una vez completado y pulsado el botón de OLÉTRIP, Ana ve las rutas recomendadas de viajes similares.



4.2 EJEMPLO DE USO. INICIO

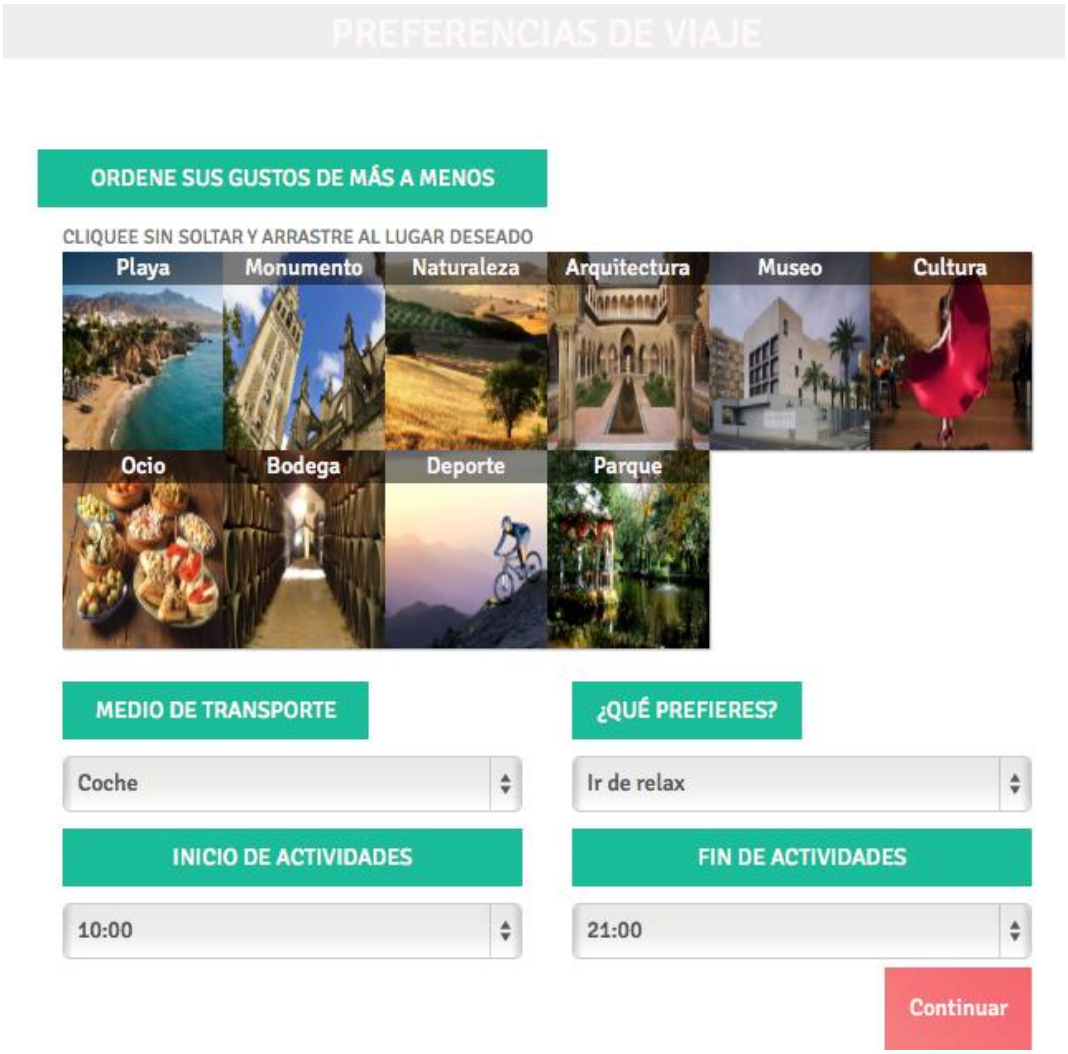
Ana empieza a explorar rutas con diferentes provincias que le resulten más interesante, finalmente elige visitar Sevilla y Málaga para su viaje de dos días.



4.3 EJEMPLO DE USO. RECOMENDACIÓN

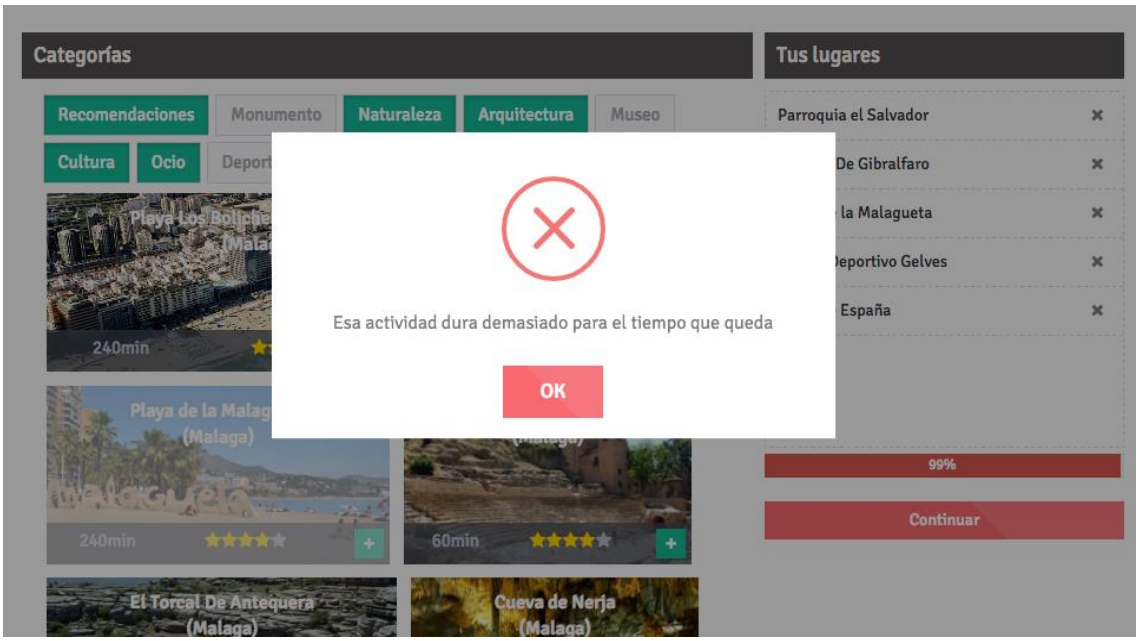
Viendo que las rutas similares recomendadas visitan varias provincias, y Ana sólo le interesa ir a Sevilla y Málaga, ya que quiere un viaje relajado con suficiente tiempo dedicado a cada actividad, decide registrarse para planificar un viaje personalizado.

Después de registrarse, Ana empieza a crear su propia ruta de viaje rellenando el formulario de preferencias que aparece al completar los datos iniciales del viaje.



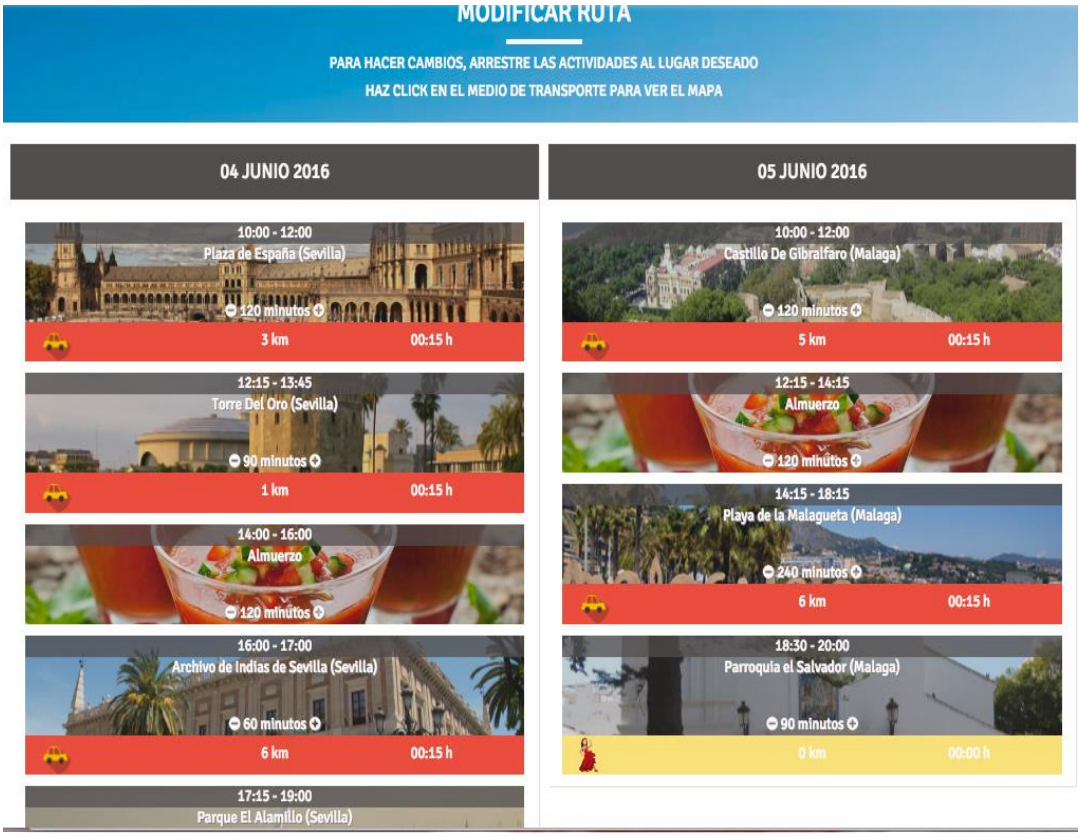
4.4 EJEMPLO DE USO. PREFERENCIAS

Una vez introducidas las preferencias sobre el viaje, Ana pulsa el botón de Continuar para visualizar las actividades que le recomienda Olétrip y seleccionar las que más le gusten. Para ello, Ana va escogiendo las actividades que le resultan más llamativas, y estás se van añadiendo al panel de "Tus Lugares". Al no saber exactamente el tiempo que se emplea para cada actividad, Ana coge demasiadas actividades y sobrepasa la cantidad de actividades dentro del tiempo empleado para el viaje.



4.5 EJEMPLO DE USO. LUGARES

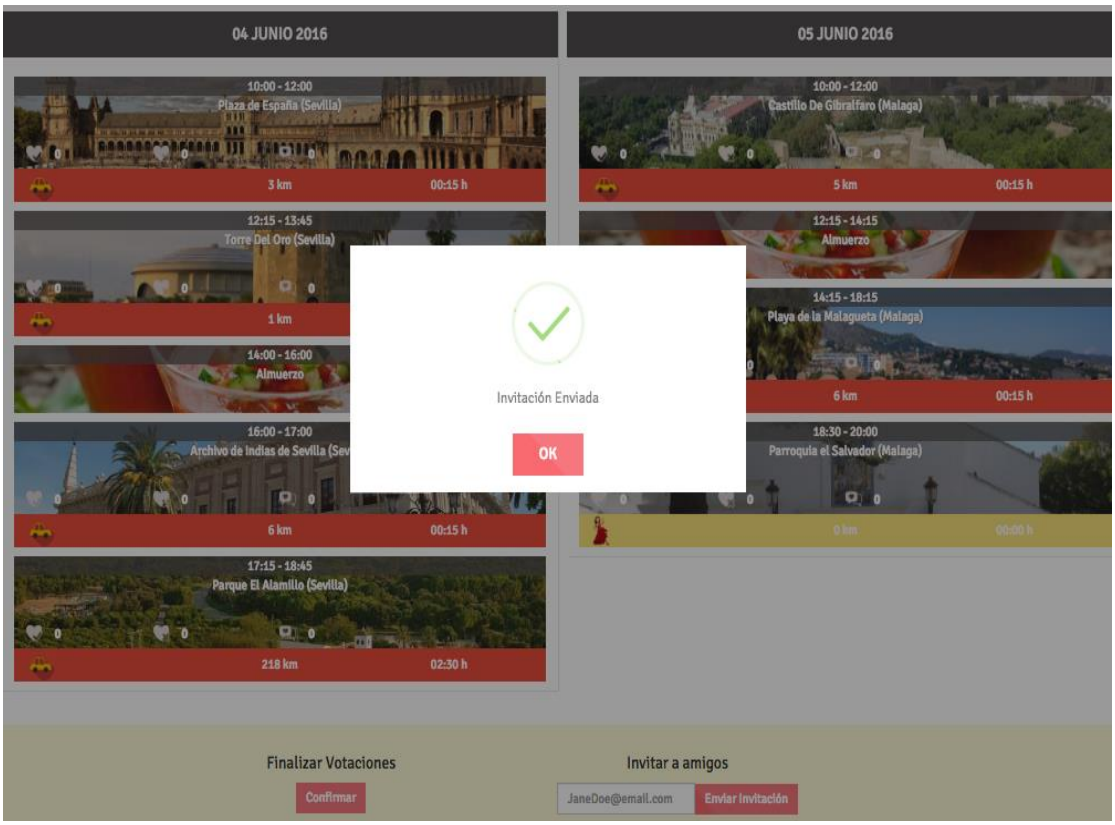
Sin embargo, el sistema le avisa del caso, e impide que sobrepase el límite establecido. Tras decidir qué actividades realizar, y pulsado el botón de "Continuar". El sistema calcula y recomienda una planificación con las actividades seleccionadas.



4.6 EJEMPLO DE USO. RUTA

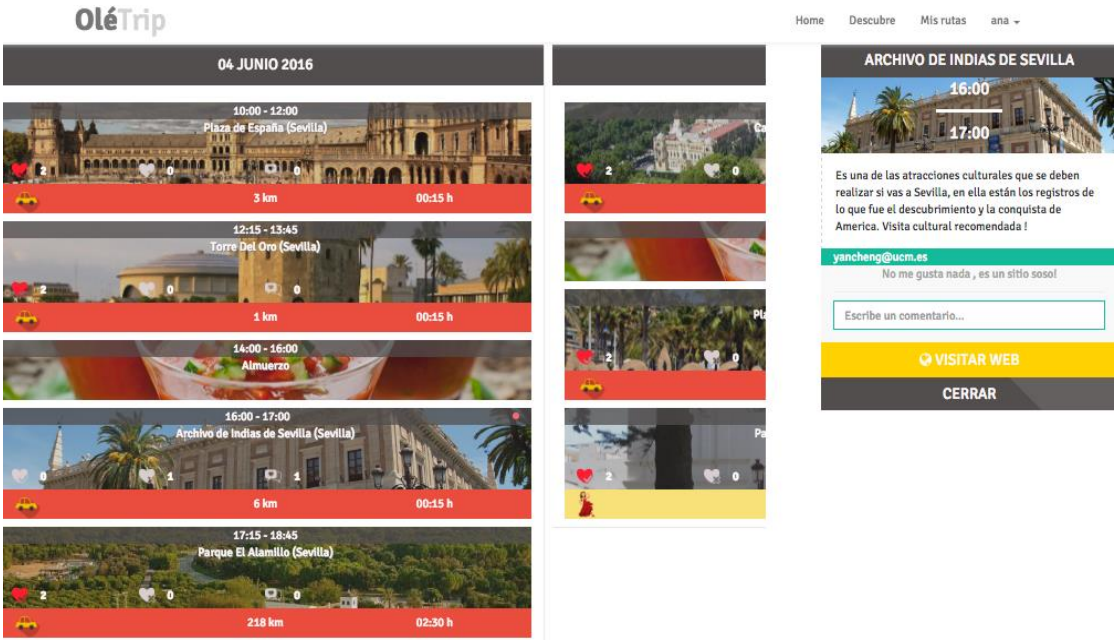
Como el plan recomendado no se ajusta mucho a lo deseado, Ana decide añadir una hora más para visitar la Plaza de España e intercambia el orden de visitas entre la Playa de la Malagueta y la Parroquia el Salvador, y así hasta conseguir lo que desea.

Después de las modificaciones sobre el plan, Ana guarda esta ruta de viaje y empieza a invitar sus amigos para participar en el proceso de valoración de la ruta.



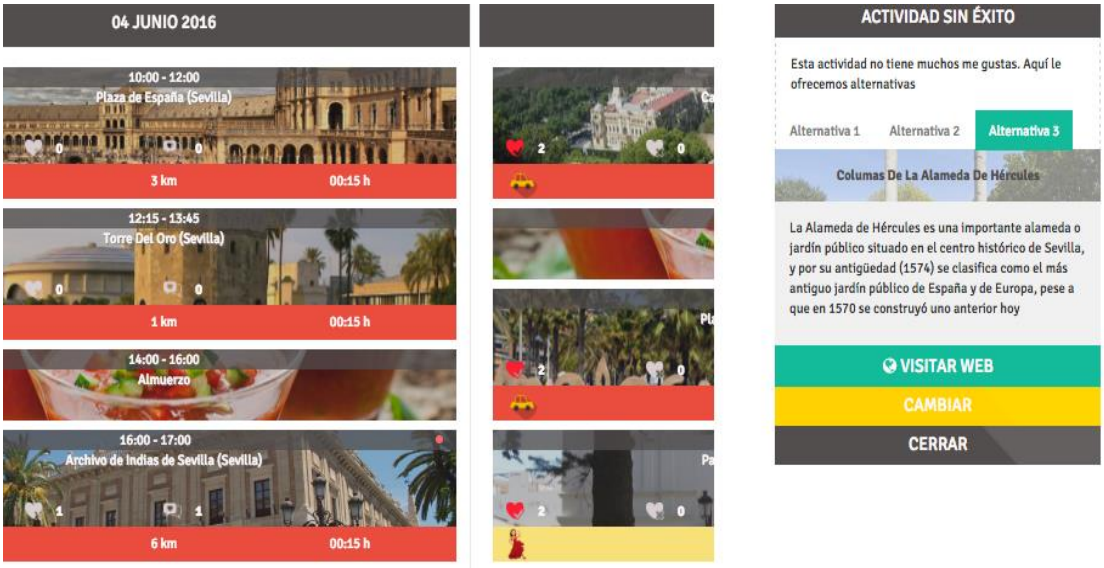
4.7 EJEMPLO DE USO. INVITACIONES

Días más tarde, Ana entra en Olétrip para ver las votaciones de sus amigos. Para ello, Ana accede al menú "Mis rutas" para seleccionar el viaje. Dentro de la planificación, ve que hay varios votos positivos y un voto negativo en la actividad Archivos de Indias de Sevilla con un comentario dejado por el invitado. Ana pulsa sobre el icono de comentario para visualizar el contenido del comentario.



4.8 EJEMPLO DE USO. DETALLES

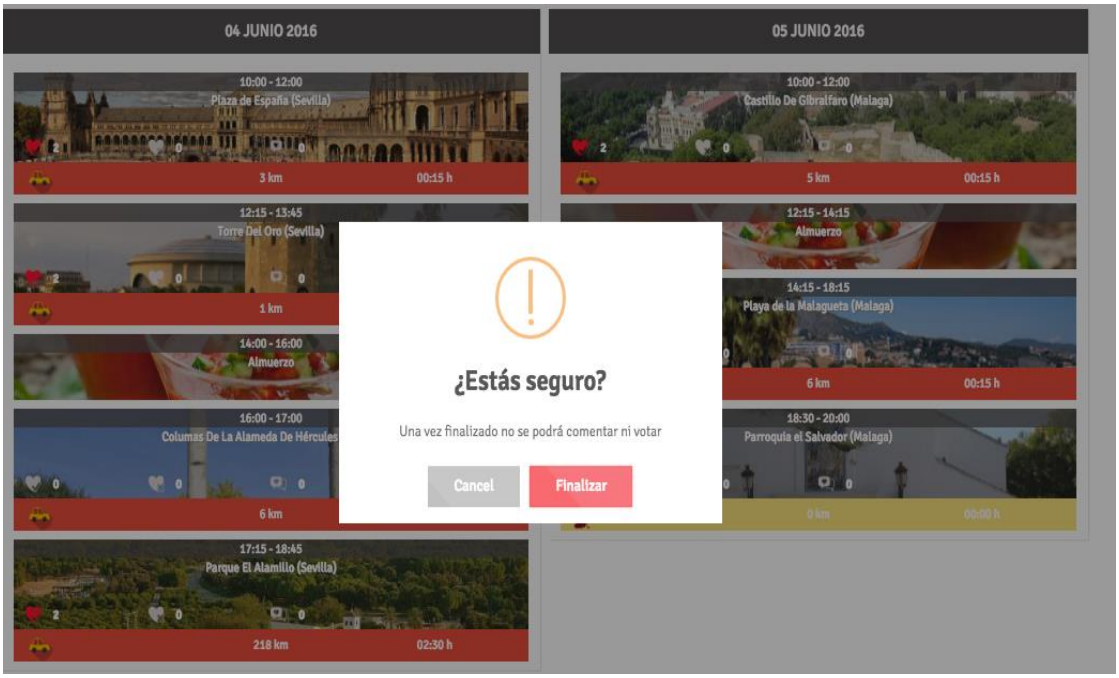
Finalmente, tras valorar el comentario y las votaciones, decide cambiar esta actividad por otra actividad alternativa y pulsa sobre el botón rojo situado en la esquina superior izquierda de dicha actividad para seleccionar alternativas.



4.9 EJEMPLO DE USO. ALTERNATIVA

Al ver las descripciones de las tres alternativas que recomienda el sistema, Ana elige la tercera alternativa Columnas de La Alameda de Hércules para sustituir a la actividad con votos negativos.

Al ver que ya no hay más actividades con votos negativos, Ana finaliza el proceso de valoración sobre la ruta pulsando el botón "Finalizar votaciones". De forma que, la planificación pasa a ser definitiva y no permite más modificaciones.



4.10 EJEMPLO DE USO. FINALIZAR

Finalmente, Ana y sus amigos realizan el viaje, y ella, días más tarde visita la web, y decide valorar el viaje, y lo puntúa con un 3, fue bastante bueno, pero mejorable. Gracias a esa puntuación, futuros usuarios, podrán obtener una mejor recomendación según sus preferencias, por tanto, se puede describir el sistema integro, como un sistema de recomendación colaborativo.

4.2. Casos de uso

Se definirán las funcionalidades que proporciona el sistema detallando el flujo normal de la obtención de la planificación. En éste caso, la aplicación estará compuesta por cuatro partes, las cuales serán detalladas a continuación.

4.2.1. Obtención de preferencias

Para que la aplicación tenga unos datos iniciales con los que poder recomendar una planificación de viaje, es necesario rellenar un formulario sencillo de preferencias para conocer algo acerca de los gustos del usuario.

Una vez que el usuario esté identificado en el sistema, podrá empezar a planificar un viaje seleccionando el/los destino/s que quiere visitar y el periodo de tiempo empleado al viaje.

Para ello tienen que proporcionarle al servicio, una serie de preferencias, en este caso categorías de viaje, donde el usuario tendrá que proporcionarlas en orden según sus gustos, es decir, de mayor a menor agrado; fijar la hora de comienzo y hora de fin para realizar las actividades, así como las provincias y el rango de fechas.

Tiene también la posibilidad de elegir el nivel de estrés de visitas, en otras palabras, si desea realizar un viaje con calma, o intentar ver todo lo posible, afectando así a la duración de las actividades.

Una vez obtenido los datos necesarios sobre el tipo de viaje, para poder ofrecer una ruta optima, también se han de proporcionar las actividades que se desean realizar, y en base a ellas, y el resto de datos, se creará una ruta personalizada, que en primera instancia no será definitiva, hasta que el usuario no lo desee, ya que será alterable.

4.2.2. Actores

En la aplicación OléTrip existen cuatro tipos de usuarios, definidos por su interactuación con la misma, y se pueden clasificar en:

- Usuario creador registrado

Tienen acceso completo a todos los servicios que ofrece la API para crear y gestionar viajes. Un usuario creador puede ser a la vez usuario invitado en otros viajes que participa.

- Usuario invitado registrado

Pueden ejercer las mismas funciones que los usuarios no registrados, además de ver las rutas a las que han sido invitado pueden dar retroalimentación a las actividades de cada viaje, es decir, votar (me gusta o no me gusta) y comentar cada una de ellas.

- Usuario no registrado

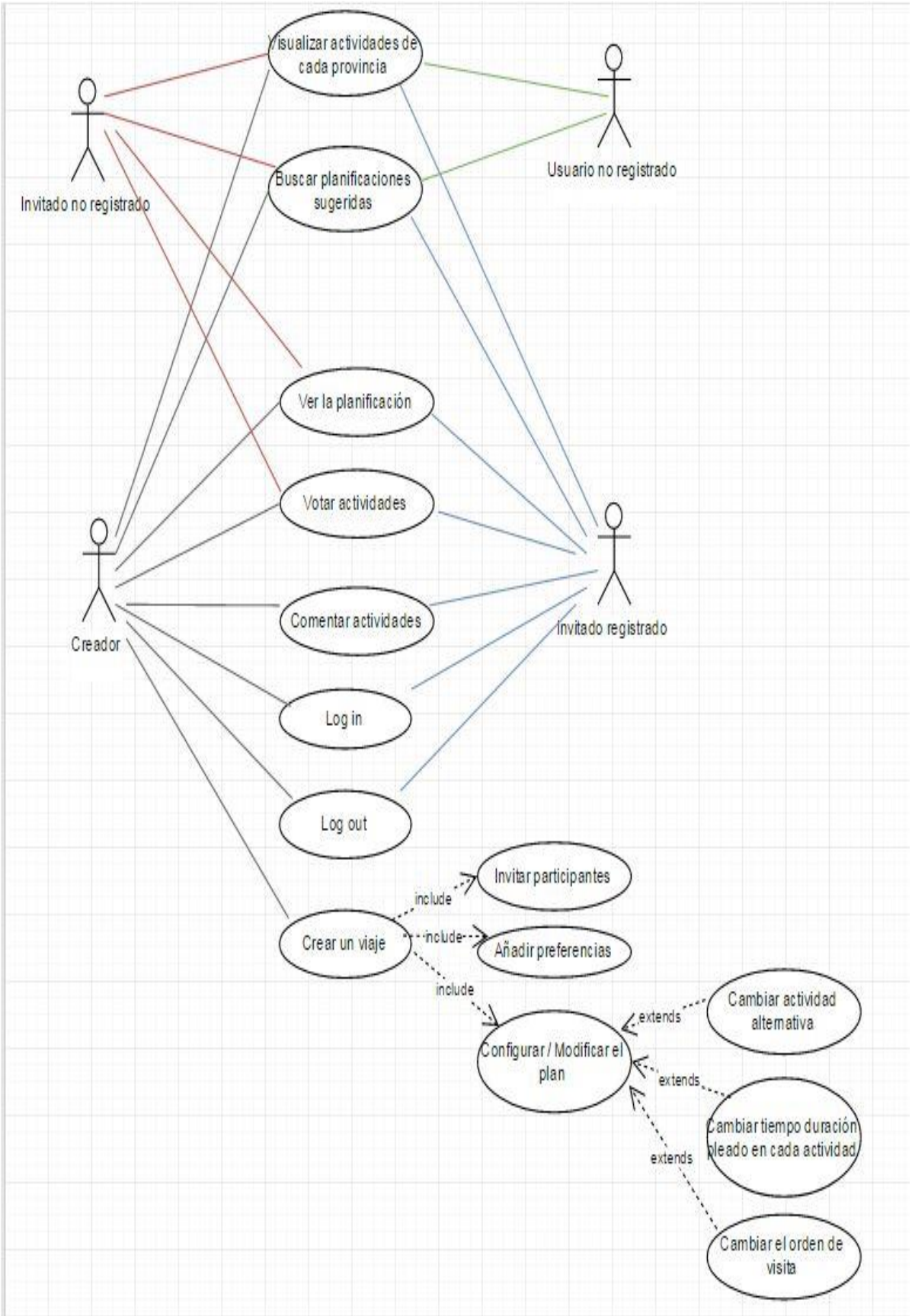
Estos usuarios sólo podrán conocer las actividades de cada lugar, y ver sugerencias de planes ya creados, si quieren crear su propio plan han de estar registrados.

- Usuario invitado no registrado

Pueden ejercer las mismas funciones que los usuarios no registrados, además de ver las rutas a las que han sido invitados, así como votar. Y si estos quieren proporcionar más retroalimentación a las actividades, es decir poder comentar, han de estar registrados.

4.2.3. Diagrama de casos de uso

Las funciones principales de cada uno de los usuarios están descritas en el diagrama de casos de uso, que se podrá ver a continuación (Ver Figura 4.11 diagrama de casos de uso)



4.11 DIAGRAMA DE CASOS DE USO

Los escenarios relevantes describen los servicios de API, pero se aplicará sobre la interfaz, de forma que reflejaría los servicios de forma más intuitiva y gráfica.

- ***Escenario crear viaje***

<table>
<tr><th>ACTOR</th><th>USUARIO CREADOR</th></tr>
<tr><td>Precondición</td><td>1. El usuario tiene que estar identificado en el sistema.
2. El usuario tiene que haber rellenado el formulario de datos iniciales del viaje y sus preferencias.
3. El usuario tiene que haber escogido un mínimo número de actividades para construir una lista de actividades.</td></tr>
<tr><td>Postcondición</td><td>1. El sistema calculará y creará una planificación del viaje teniendo las preferencias del usuario y repartir actividades por distancia de geolocalización en el rango de tiempo total del viaje.
2. El sistema mostrará una planificación del viaje con las actividades y la duración de cada una de ellas y el tiempo empleado para llegar de una actividad a la siguiente, separados en listas diarias.</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Pasos</th><th>Ejecución</th><th>Servicios</th></tr>
<tr><td>1</td><td>El sistema muestra un cuadro inicie selector de provincias, fecha inicio y fecha fin.</td><td>-</td></tr>
<tr><td>2</td><td>El usuario introduce los datos y pulsa el botón de 'Olétrip'</td><td>-</td></tr>
<tr><td>3</td><td>Aparece un formulario de preferencias sobre el viaje, donde el usuario tiene que ordenar categoría, elegir medio de transporte, grado de</td><td>-Get_category</td></tr>
</table>
</td></tr>
</table>

		estrés y franja de horarios para las actividades diarias.	
	4	El usuario pulsa el botón 'continuar'	-Create_trip -Get_sites
	5	Aparece un selector de categoría donde el usuario tiene que ir seleccionando las categorías y arrastrar las actividades que quiera realizar al 'tus lugares' situado a la derecha.	
	6	El usuario pulsa el botón 'continuar' para seguir	-Calculare_trip
	7	El sistema muestra una planificación del viaje elaborado según los datos introducidos.	
	8	El usuario pulsa el botón 'Guardar' para guardar el plan	
	9	Aparece un dialogo de confirmación, donde el usuario pulsará 'Guardar'	-Save

1 CASOS DE USO. ESCENARIO CREAR VIAJE

- ***Recomendar viajes existentes***

<table>
<tr><th>ACTOR</th><th>TODOS LOS USUARIOS</th></tr>
<tr><td>Precondición</td><td>-</td></tr>
<tr><td>Postcondición</td><td>El sistema muestra los viajes existentes similares con los datos seleccionados por el usuario</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Pasos</th><th>Ejecución</th><th>Servicios</th></tr>
<tr><td>1</td><td>El sistema muestra un cuadro de selección de provincias, fecha inicio y fecha fin.</td><td></td></tr>
<tr><td>2</td><td>El usuario introduce los datos y pulsa el botón de 'Olétrip'</td><td>-Recommend</td></tr>
<tr><td>3</td><td>El sistema muestra todos los viajes existentes relacionados, con los detalles de cada uno.</td><td></td></tr>
</table>
</td></tr>
</table>

2 CASOS DE USO. RECOMENDAR PLANIFICACIÓN

- ***Escenario invitar participantes***

<table>
<tr><th>ACTOR</th><th>USUARIO CREADOR</th></tr>
<tr><td>Precondición</td><td>1. El usuario tiene que estar identificado en el sistema.
2. El usuario tiene que haber creado una planificación del viaje para poder invitar a los participantes.</td></tr>
<tr><td>Postcondición</td><td>El sistema enviará correos electrónicos a los usuarios participantes con la url de la página de planificación del viaje.</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Pasos</th><th>Ejecución</th><th>Servicios</th></tr>
<tr><td>1</td><td>El usuario accede al menú 'Mis Rutas'.</td><td>-</td></tr>
<tr><td>2</td><td>El sistema muestra las rutas que tiene creado el usuario.</td><td>-</td></tr>
<tr><td>3</td><td>El usuario selecciona la ruta a la que quiere invitar a participantes.</td><td>-</td></tr>
<tr><td>4</td><td>El sistema muestra la planificación de la ruta y el cuadro de texto para introducir correos de los invitados.</td><td>-</td></tr>
<tr><td>5</td><td>El usuario introduce los correos y pulsa el botón 'Enviar Invitación'</td><td>-Send invitation</td></tr>
</table>
</td></tr>
</table>

3 CASOS DE USO. ESCENARIO INVITAR PARTICIPANTES

- ***Escenario votar actividad***

<table>
<tr><th>ACTORES</th><th>USUARIO INVITADO REGISTRADO
USUARIO INVITADO NO REGISTRADO</th></tr>
<tr><td>Precondición</td><td>El usuario tiene que haber recibido la invitación por parte del usuario creador sobre el viaje concreto y acceder a la planificación a través del link generado automáticamente.</td></tr>
<tr><td>Postcondición</td><td>1. El sistema actualiza y muestra el estado de voto de la actividad después de cada votación del usuario.
2. El sistema aumentará el contador de ‘likes’ o ‘dislikes’ según el voto realizado por el usuario.</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Pasos</th><th>Ejecución</th><th>Servicios</th></tr>
<tr><td>1</td><td>El usuario entra en la página de visualización de planificación del viaje a través de la url recibida.</td><td>-Get_trip_info</td></tr>
<tr><td>2</td><td>El usuario puede votar 'Me gusta' o 'No me gusta' en cada actividad</td><td>-Route_comments
-Get_trip_activity_likes
-
Get_trip_activity_dislikes</td></tr>
<tr><td>3</td><td>El sistema actualiza los dislikes y likes de cada actividad</td><td></td></tr>
</table>
</td></tr>
</table>

4 CASOS DE USO. ESCENARIO VOTAR ACTIVIDAD

- ***Escenario comentar actividad***

<table>
<tr><th>ACTOR</th><th>USUARIO INVITADO REGISTRADO</th></tr>
<tr><td>Precondición</td><td>1. El usuario tiene que haber recibido la invitación por parte del usuario creador sobre el viaje concreto y accede a la planificación a través del link generado automáticamente.

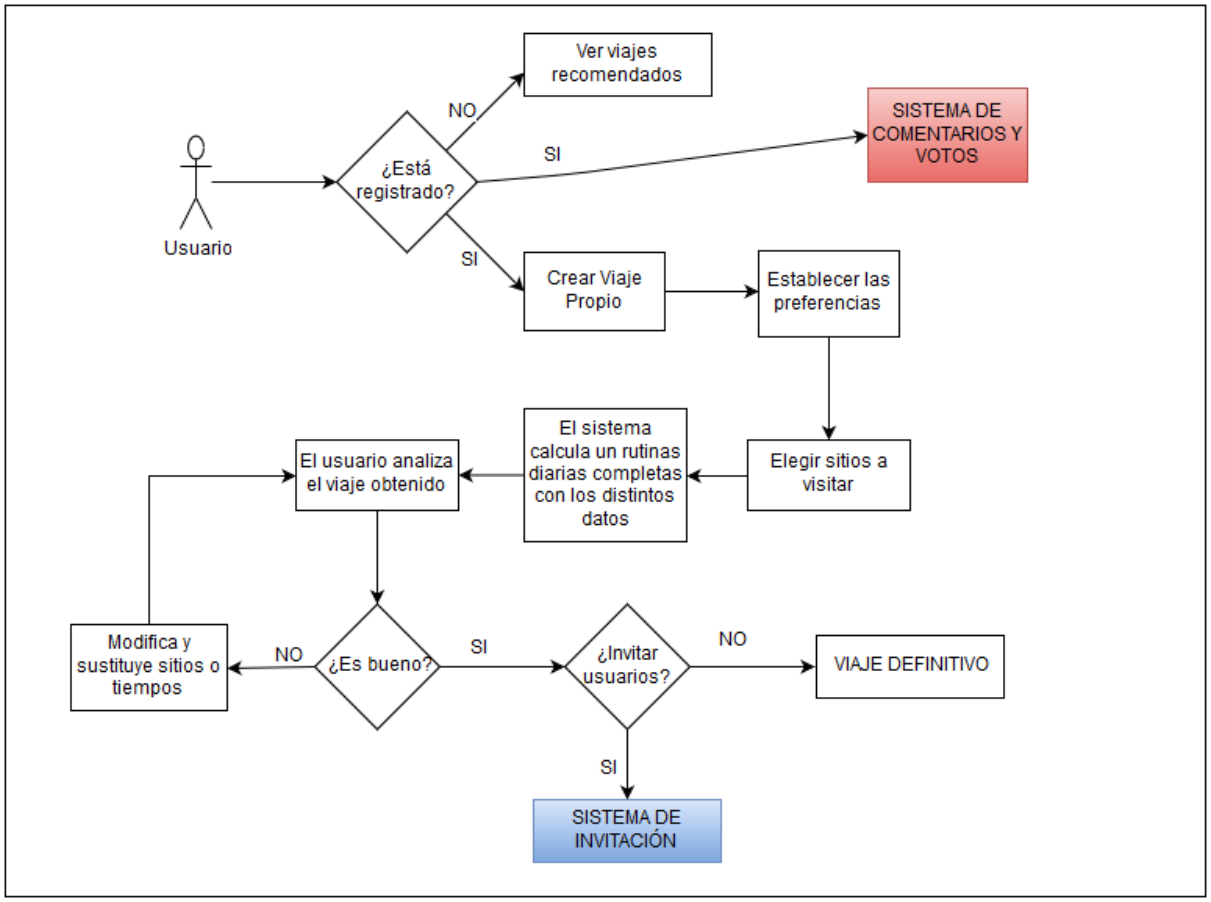
2. El usuario tiene que haberse logueado en el sistema para poder realizar comentarios sobre la actividad.</td></tr>
<tr><td>Postcondición</td><td>El sistema actualizará y mostrará el comentario realizado por el usuario en la página de planificación del viaje, con la información del nombre del autor, fecha y hora del comentario y el cuerpo del mensaje.</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Pasos</th><th>Ejecución</th><th>Servicios</th></tr>
<tr><td>1</td><td>El usuario entra en la página de visualización de planificación del viaje a través de la url recibida.</td><td>-Get_trip_info</td></tr>
<tr><td>2</td><td>El usuario puede comentar sobre cada actividad</td><td>-
Get_trip_activity_comments</td></tr>
<tr><td>3</td><td>El sistema desliza un panel con la descripción de la actividad y un cuadro para introducir comentario</td><td></td></tr>
</table>
</td></tr>
</table>

4	El usuario introduce el comentario y lo envía	-Put_comments
5	El sistema actualiza el panel con el comentario y su autor	-Get_trip_activity_comments

5 CASOS DE USO. ESCENARIO COMENTAR ACTIVIDAD

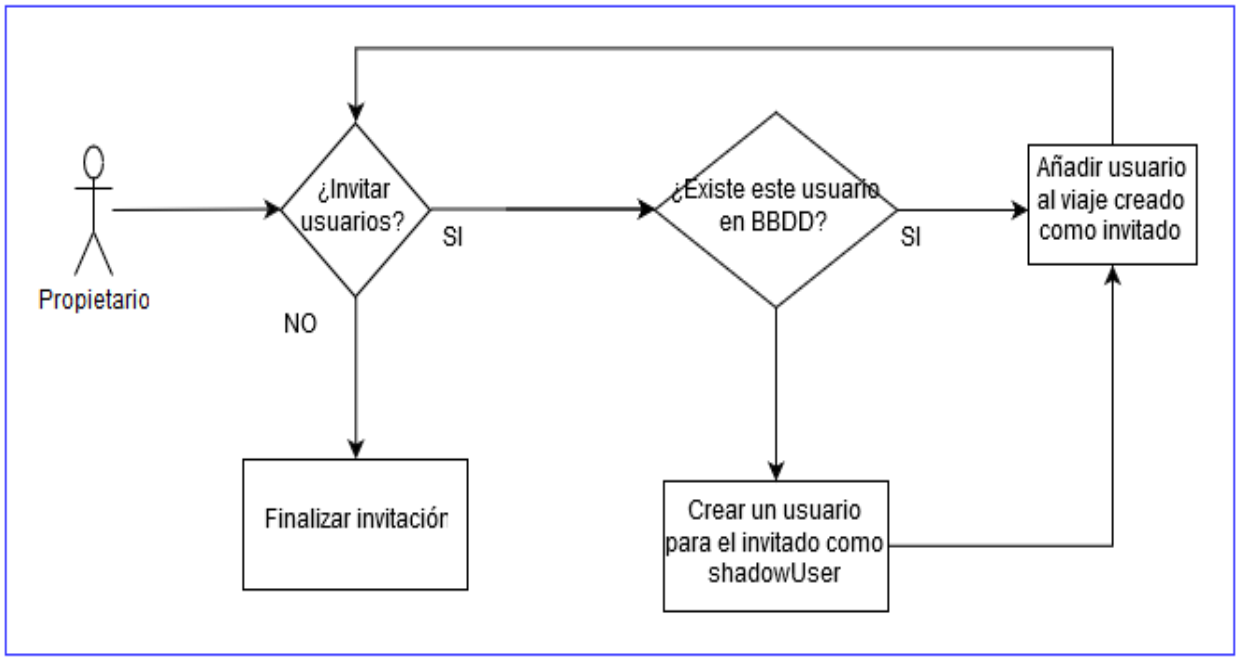
4.2.4. Diagrama de flujos

- *Diagrama de flujos del camino lógico para un usuario*



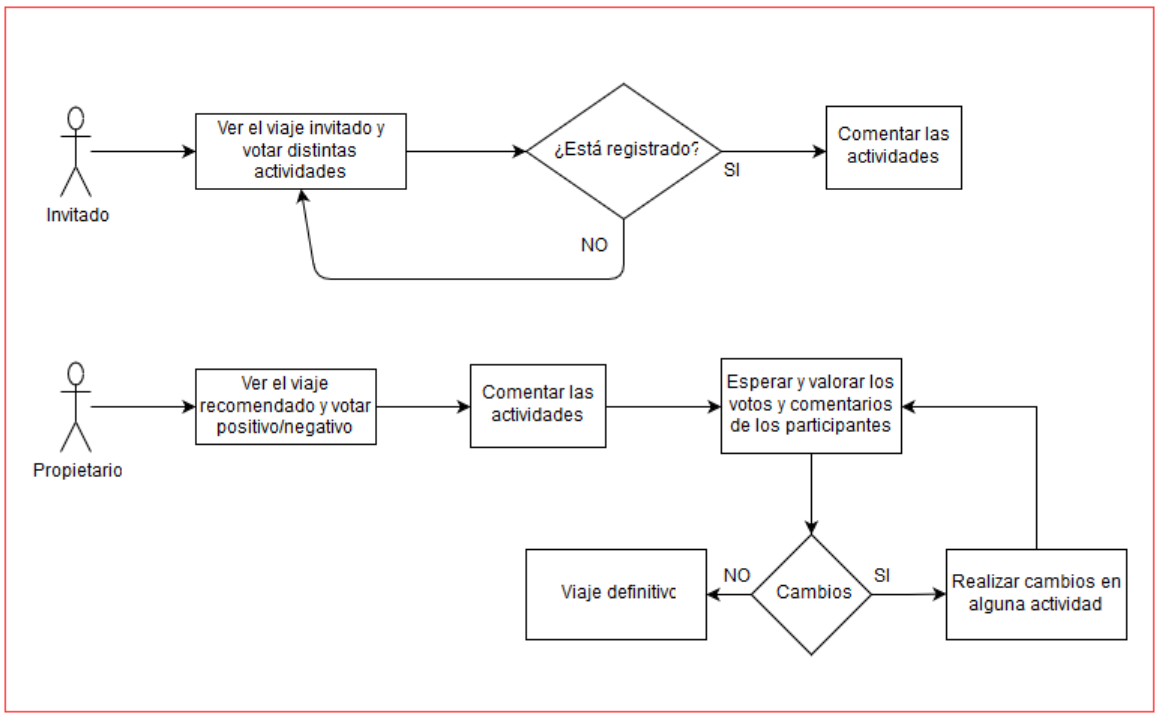
4.12 DIAGRAMA DE FLUJOS. USUARIO

- *Diagrama de flujos para el sistema de invitaciones*



4.13 DIAGRAMA DE FLUJOS. INVITACIONES

- *Diagrama de flujos para el sistema de votos y comentarios*



4.14 DIAGRAMA DE FLUJOS. RETROALIMENTACIÓN

5. Arquitectura del sistema

En esta fase se han elaborado varios apartados donde se especificarán los posibles servicios a construir para cubrir todas las necesidades del API, así como las funcionalidades ofrecidas por la interfaz web. Por otra parte, se han realizado también distintos bocetos tanto como en papel o en formato digital de las distintas interfaces e interacciones entre estas, en la parte de diseño de base de datos se ha empleado bastante tiempo para poder llegar a obtener finalmente un diseño decoroso y consistente.

5.1. Arquitectura Modular

Se ha diseñado una arquitectura Cliente – Servidor. Esta arquitectura es muy usada ya que destaca por estar separada físicamente en dos agentes básicos de intercambio de información: el cliente y el servidor. Estos dos agentes son independientes entre sí desde un punto de vista lógico, y trabajan a través de una red de comunicaciones para realizar una o varias tareas. El cliente realizará peticiones a un servicio y recibirá la respuesta de esta petición, en cambio el servidor recibe y procesa la petición, devolviendo la respuesta solicitada.

5.1.1. Características

- **Los agentes están débilmente acoplados.**

  El cliente y el servidor son sistemas independientes y separados físicamente, que solamente intercambian solicitudes de servicios y respuestas en JSON, al cliente le es trasparente el funcionamiento interno del API y al servidor le es trasparente el manejo de los datos proporcionados mediante respuestas.

- **Protocolos asimétricos y recursos**

  Entre cliente y servidor se establece una relación de N a 1, esto quiere decir que un servidor puede atender a muchos

clientes al mismo tiempo. Siempre será un cliente quien inicia el dialogo.

- **Independencia de tecnologías y lenguajes**

  El desarrollo de ambos agentes no está ligado, esto permite que se pueda desarrollar los distintos agentes según necesidad, o que resulte más cómodo a la hora de trabajar. Cabe destacar que podemos utilizar el mismo servidor para distintas aplicaciones de distintas plataformas como: aplicaciones Android, aplicaciones IOS, páginas web o incluso ser utilizada para otras APIs.

- **Gran escalabilidad, flexibilidad y fiabilidad.**

  Por parte del servidor se pueden implementar nuevas funcionalidades o mejorar las existentes sin tener impacto en el otro agente, y siempre manteniendo el mismo formato de respuestas y el flujo de peticiones. También se puede aumentar o migrar a máquinas servidoras más potentes para proporcionar servicios a muchos más clientes al mismo tiempo.

  A esto se le puede sumar la trasparencia de ubicación, un servidor puede estar en el mismo equipo que el cliente o en otra red.

- **Experiencia del usuario**

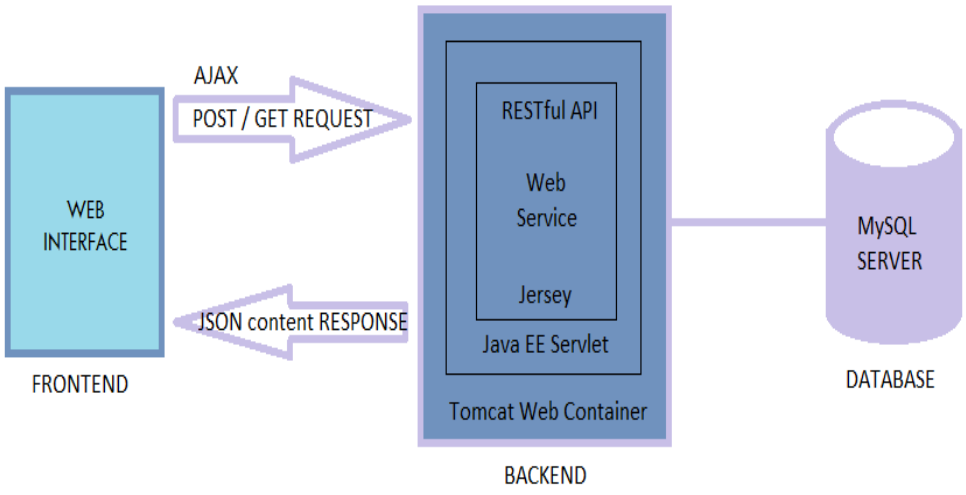
  Las respuestas del servidor son datos planos en formato JSON, por lo que el tiempo de transferencia es mucho menor que la petición de una página completa. Junto con el uso de AJAX conseguimos también que estas aplicaciones se asemejen más a aplicaciones de escritorio sin necesidad de refrescar la página.

5.1.2. Arquitectura Cliente – Servidor (multicapas)

Dentro de esta arquitectura se ha aplicado una arquitectura de dos capas dentro de esta estructura. Esto permite la constante evolución para satisfacer los nuevos requisitos o mejoras.

La primera capa de presentación corresponde al cliente, ya que este se encarga de representar y mostrar los datos obtenidos de las respuestas de las peticiones.

La segunda capa viene subdividida en dos capas lógicas: una que consta de la lógica de negocio, del cual se encarga de procesar las peticiones entrantes, trabajos de cómputos y elaboración de respuestas en formato JSON, y otra que se encarga de la administración de la base de datos, para dicho acceso se ha emplea el patrón de acceso de datos (DAO).

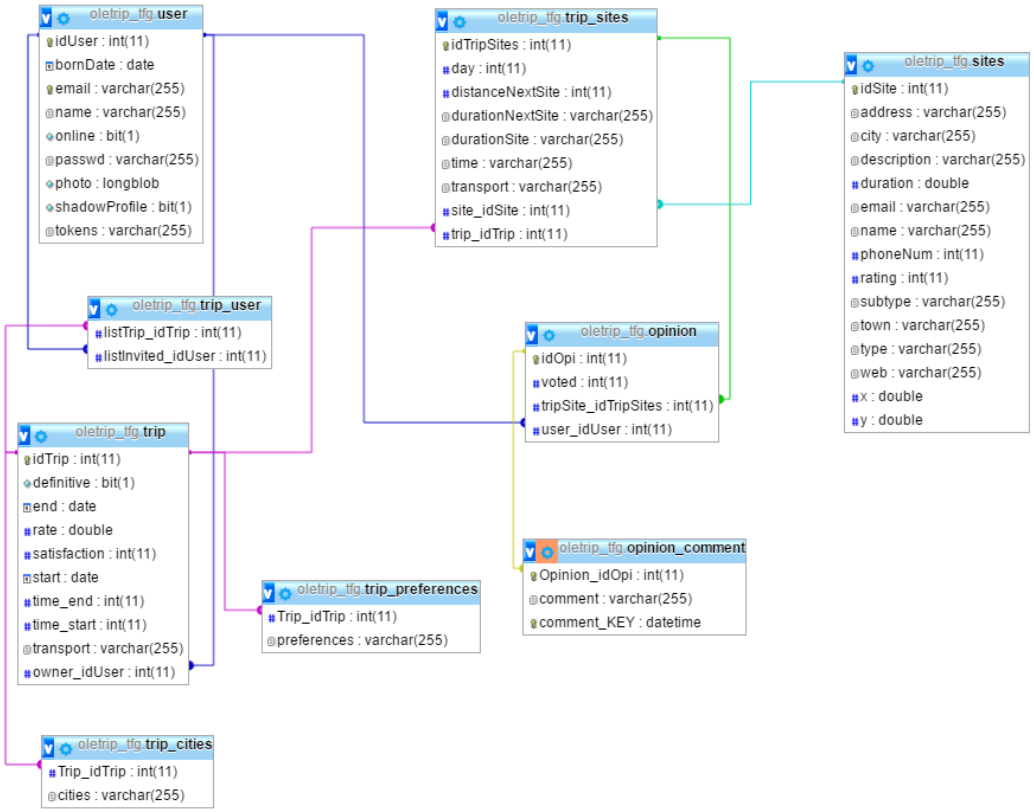


5.1 ARQUITECTURA MODULAR

5.2. Base de datos

Para guardar toda la información se ha diseñado una base de datos relacional, la cual ha sufrido diversas modificaciones con constantes cambios para dar como resultado, una base de datos más segura, simple y descriptiva.

El diseño final de las tablas queda reflejado en la siguiente figura, y la estructura de las tablas se detallará en el **Apéndice 9.**



5.2 BASE DE DATOS. ESQUEMA

- ***Usuario***

Esta tabla guarda tanto información del perfil del usuario como datos sobre la cuenta del usuario.

Cabe destacar, el campo online, una celda booleana que permite controlar si el usuario ha iniciado sesión o está desconectado, si un usuario no inicia sesión previamente no se le permite realizar algunas interacciones con la aplicación (crear un viaje, comentar, etc.).

Por otra parte, el campo shadowProfile, es una celda booleana que índica si un usuario pertenece de forma validada al sistema o no. Éste último caso ocurre cuando se envía una invitación del viaje a un usuario, y al no estar

registrado, se añade para un futuro registro y controlar que sea un usuario perteneciente al viaje.

Esta tabla tiene una relación N a N con la tabla Viajes, es decir, un usuario puede crear varios viajes y a la vez ser partícipe de muchos.

- ***Viajes***

La tabla Viajes guarda información sobre el viaje relacionada con el viaje (días, ciudades…). Mantiene una relación N a N con la tabla Usuarios.

Cada viaje será creado por un usuario identificado del sistema, y a su vez puede tener muchos usuarios que participe en él. En dicho viaje se guardarán las distintas configuraciones realizadas por el creador, tales como la lista de preferencias, la lista de ciudades a visitar, la lista de participantes, días que comprende el viaje, el transporte, etc...

Un viaje contendrá muchos sitios, los cuales estarán organizados por día y hora, por cada actividad se guardará, también, la duración empleada para visitar dicho sitio, el transporte que se ha decidido emplear para llegar allí y datos relacionados con la distancia y tiempo empleado en el transporte. Toda esta información será guarda en una tabla intermedia Sitios-Viaje que relaciona tabla Viajes con la tabla Sitios.

Cuando un usuario creador finaliza el proceso de configuración y valoración, se podrá cambiar el estado del viaje a definitivo, esto se verá reflejado en el campo booleano Definitive, y no se permitirá ninguna modificación sobre el viaje.

El campo Satisfaction se obtiene de la valoración general sobre la ruta realizada, siendo positiva puede ser utilizada posteriormente para ser recomendada a otros usuarios. Este dato se obtiene a través del creador del viaje, el cual lo calificará en una escala de 0-5.

- ***Sitios***

Esta tabla, en primera instancia, debía ser una derivación de una base de datos proporcionada por una empresa del sector turístico, pero dicha base de datos se descartó finalmente porque había muchos datos innecesarios y se decidió construir una nueva, salvando algunos datos ofrecidos por tal base de datos.

Como se ha mencionado en la página anterior, la tabla Sitios tiene una relación N a N con la tabla Viajes. Ya que un viaje contiene muchos sitios a visitar, y a la vez, estos sitios pueden ser incluidos en diferentes viajes.

La tabla Sitios contiene información representativa de los diferentes lugares que se pueden visitar, como son las coordenadas geográficas, tipo, descripción, página web sobre el lugar, etc...

Hay que mencionar que para cada sitio se ha buscado también una imagen representativa, dichas imágenes están guardadas por separado motivos de tamaño. Estas imágenes se encuentran alojadas en el mismo servidor que la interfaz (oletrip.esy.es), donde mediante el identificador del sitio se puede acceder a tal imagen.

- ***Opiniones***

En esta tabla se guardan las valoraciones de cada actividad de un viaje. Tiene una relación ternaria de Uno a Uno a N en tablas Usuario, Opinion y Sitios-Viaje respectivamente, es decir, que cada participante o creador de un viaje podrá realizar una opinión sobre las distintas actividades contenida en ella.

Para ello existe un sistema de votación de positivos y negativos (me gusta y no me gusta) y también la posibilidad de escribir comentarios (tipo foro). Esto es esencial, ya que para que el viaje sea lo más satisfactorio para todos los integrantes, se necesita tener tal información. Finalmente, dichas opiniones y/o votaciones afectarán a la decisión final del creador, el cual podrá optar por cambiarlos y/o ajustarlos.

5.3. Servicios

Como se ha comentado, Olétrip es una aplicación accesible desde cualquier plataforma, ya sea móvil, PC, etc., donde el módulo de servicios es el encargado de que esto sea posible. El cual está compuesto por diversos servlets, que proporcionan diferentes servicios y/o funcionalidades.

Para interactuar con el sistema se han de realizar llamadas a la url correspondiente, y con los parámetros correspondientes, un ejemplo de url sería el siguiente:

```
http://<server>/TFG_OLETRIP/trip/1
```

Los parámetros se han de proporcionar, algunos mediante la url, o mediante formato JSON, y las respuestas de cada servicio siempre serán devueltas en el mismo formato. Tal respuesta puede ser un error (falta de permisos, parámetros erróneos …), o por el contrario una respuesta exitosa con los datos obtenidos.

A continuación, se describirán los servicios con los que cuenta el sistema, y posteriormente en el **Apéndice B** se detallarán en profundidad, explicando cómo se utilizaría cada servicio, con los parámetros necesarios, y la respuesta de los mismos.

- ***User Services***

Servicios relacionados con la gestión de usuarios.

- ***Registro***

  Método que permite al usuario registrarse en la aplicación, proporcionando los datos necesarios para ello, es decir, nombre, email, contraseña y fecha de nacimiento. [`/user/signup`]

- ***Login***

  Una vez registrado en el sistema, para poder utilizar varios servicios, es necesario estar logueado, para ello el usuario ha de proporcionar sus

email y contraseña, y para que el login sea correcto, los datos proporcionados han de coincidir con los de la base de datos del sistema. [`/user/login`]

- ***Datos de usuario***

  Método que devuelve los datos asociados a un usuario, omitiendo los tokens y contraseña del mismo. [`/user/{id_user}`]

- ***Modificación de datos de usuario***

  Permite modificar la contraseña, fecha de nacimiento, nombre y contraseña de usuario. Comprobando que sean datos válidos. [`/user/{id_user}/update`]

- ***Cierre de sesión***

Destruye la sesión, evitando así el uso de varios servicios. [`/user/logout`]

- ***Trip Services***

Servicios relacionados con la gestión de los viajes de los usuarios.

- ***Creación de un viaje***

  Método inicial para crear la ruta deseada, donde se le proporcionan las preferencias del mismo. [`/trip/create`]

- ***Creación de la ruta***

  Se le proporcionan las actividades deseadas, y éste las procesa para devolver la ruta idónea para poder visitarlas. [`/trip/{id_rute}/rute`]

- ***Guardar la ruta***

  Método que permite la salvación de los datos de la ruta en la base de datos. [`/trip/{id_rute}/save`]

- ***Recuperación de ruta***

  Permite ver los datos guardados referentes a una ruta, desde los días y ciudades, a las actividades en el orden de visita. [`/trip/{id_rute}`]

- ***Ruta definitiva***

Método que permite que una ruta sea definitiva, evitando así cualquier modificación en la misma. [`/trip/{id_rute}/definitive`]

- ***Creador***
  Permite saber quién es el creador de la ruta. [`/trip/{id_rute}/owner`]

- ***Invitar***
  Crear y envíar invitaciones a los usuarios, siempre y cuando esta acción la ejecute el creador de la ruta. [`/trip/{id_rute}/invite` y `/trip/{id_rute}/invited/{id_user}/send`]

- ***Invitados***
  Se han realizado dos métodos que permiten saber los invitados de un viaje, o saber si un usuario está invitado. [`/trip/{id_rute}/invited`]

- ***Valoración del viaje***
  Se permite valorar un viaje, y saber cual tiene, restringiendo la valoración solo al creador del mismo. [`/trip/{id_rute}/vote`]

- ***Eliminar un viaje***
  Si un creador decide que ya no quiere un viaje, o ya pasó, éste podrá eliminarlo mediante este método. [`/trip/{id_rute}/delete`]

- ***Eliminar invitación***
  Existen dos formas de poder eliminar una invitación, que el creador te elimine de la lista de invitados, o que el propio invitado decida salir, para ello se emplea este método, comprobando que el que ejecuta la acción sea uno de los usuarios nombrados. [`/trip/{id_rute}/user/{id_user}/noinvite`]

- ***Recomendación de viajes***
  Sin estar registrado se pueden obtener una serie de recomendaciones de rutas, basadas en viajes de usuarios pertenecientes a la aplicación. [`/trip/recommend`]

- ***Activity Services***

Servicios relacionados con la gestión de las actividades de las rutas.

- ***Resumen de actividad***

  Devuelve todo lo referente a una actividad concreta (horario, duración, ubicación, etc.) [`/trip/{id_rute}/activity/{id_act}`]

- ***Votación de actividad***

  Método que permite definir el agrado por una actividad concreta de una ruta, es decir, si le gusta o no. [`/trip/{id_rute}/activity/{id_act}/vote`]

- ***Comentarios***

  Se permite recuperar comentarios de una actividad concreta, así como comentar en ella, siempre y cuando sea el creador del viaje o un invitado. [`/trip/{id_rute}/activity/{id_act}/comments`]

- ***Voto de un usuario***

  Permite conocer la votación de un usuario concreto. [`/trip/{id_rute}/activity/{id_act}/user/{id_user}/vote`]

- ***Me gusta y no me gusta***

  Devuelve el número de me gusta o no me gustas de una actividad en concreto. [`/trip/{id_rute}/activity/{id_act}/like` o `/trip/{id_rute}/activity/{id_act}/dislike`]

- ***Alternativa a una actividad y modificación***

  Si una actividad no tiene muchos me gustas puede ser cambiada por otra, que se devolverá según sus preferencias de viaje al usuario creador. [`/trip/{id_rute}/activity/{id_act}/alternative`]

- ***Site Services***

Servicios relacionados con los lugares con los que cuenta la aplicación.

- ***Información de un lugar***

  Devuelve todo lo relativo a un lugar concreto almacenado en la base de datos. [`/site/{site_id}`]

- ***Lugares pertenecientes a una provincia***

Muestra todos los lugares que pertenecen a x provincia. [`/site/city/{provincia}/`]

- ***Lugares pertenecientes a una provincia y categoría.***

Dado una provincia y categoría devuelve los lugares que pertenezcan ellas. [`/site/city/{provincia}/category/{categoria}`]

- ***Lugares pertenecientes a una categoría***

Según x categoría se devuelve los lugares que son de dicha categoría. [`/site/city/category/{categoria}`]

- ***Categorías de una provincia***

Según x provincia se devuelve las categorías con la que ésta cuenta. [`/site/{provincia}/categories`]

- ***Other Services***
  - ***Categorías***

    Devuelve las categorías con las que cuenta la aplicación (Arquitectura, Parque, Playa, Ocio, Cultura, Bodega, Monumento, Museo, Naturaleza). [`/service/category`]

  - ***Provincias***

    Devuelve las provincias con las que cuenta la aplicación. [`/service/cities`]

5.4. Interfaz

Para poder mostrar el uso de la API Olétrip, hemos diseñado una interfaz de usuario que permite acceder a los servicios creados, mediante una aplicación web. Sin embargo, la finalidad de la API no es sólo para uso exclusivo de esta interfaz, sino que podrá utilizarse con otras aplicaciones, pero esta es una forma más visual de ver el funcionamiento.

La interfaz web está implementada con HTML5 y JavaScript, ya que estos nos proporcionan elementos que facilitan la programación, diseño y manejo de datos, así como el ahorro de trabajo. Además, empleando Bootstrap como soporte de diseño, que permitirá que la interfaz sea usable en múltiples dispositivos (adaptative design). Además, cabe destacar que antes de realizar la implementación se realizaron una serie de mockups que permitió consolidar una idea de diseño, estos podrán encontrarse en el **Apéndice C**, sin embargo estos no son una representación del diseño final, ya que sufrió modificaciones a lo largo del desarrollo de la interfaz.

5.3 INTERFAZ. RESPONSIVE DESIGN

Para la interacción con la API OléTrip, se ha empleado AJAX como mecanismo de comunicación, enviando y recibiendo datos tratados mediante JSON, los cuales serán procesados y mostrados al usuario, sin necesidad de procesos tediosos.

Como ya se ha comentado, existen 2 tipos de usuarios, registrados y no registrados, y dentro de ellos otros tipos de usuarios:

- Usuarios registrados
  - Creador
  - Invitado
- Usuarios no registrados
  - No invitado
  - Invitado

A continuación, se detallarán los diferentes servicios que ofrece la interfaz. Cabe destacar que todas las llamadas a los servicios, como se ha comentado, se realizan con AJAX, de la siguiente manera:

- ***Servicios POST***

```
datos = JSON.stringify(elementos);
$.ajax({
    url: URL_API + SERVICIO
    data: datos,
    type: "POST",
    async: false,
    contentType: "application/json",
    dataType: "json",
    success: function(data) {
        ...
    }
  });
```

- ***Servicios GET***

```
$.ajax({
    url: URL_API + SERVICIO,
    type: "GET",
    async: false,
    success: function(data) {
      …
    }
});
```

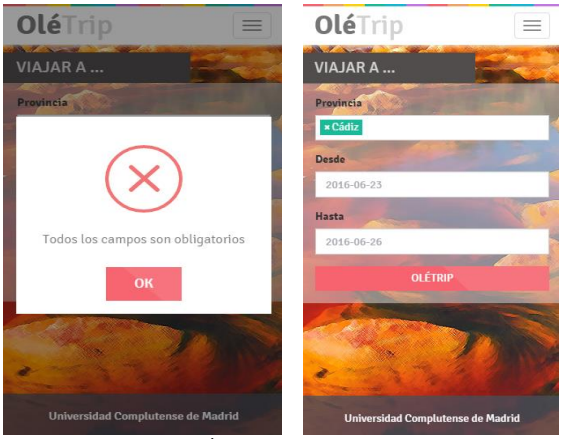
- ***Servicios comunes***

Como parte común todos los usuarios podrán acceder:

- ***Página de inicio***

  En la página principal el usuario puede empezar a planificar el viaje, decidiendo las provincias deseadas, y los días deseados, para ello

contará con un formulario que lo llevará a las preferencias o a las sugerencias, dependiendo del tipo de usuario (registrado o no registrado). Antes de que este sea enviado se comprueba que todos los campos son válidos, que no sean vacíos y que la fecha de inicio no sea posterior a la de finalizar el viaje. En cualquier caso, erróneo, serán notificados al usuario.



5.4 INTEFAZ. PÁGINA DE INICIO

- ***Página de descubre***

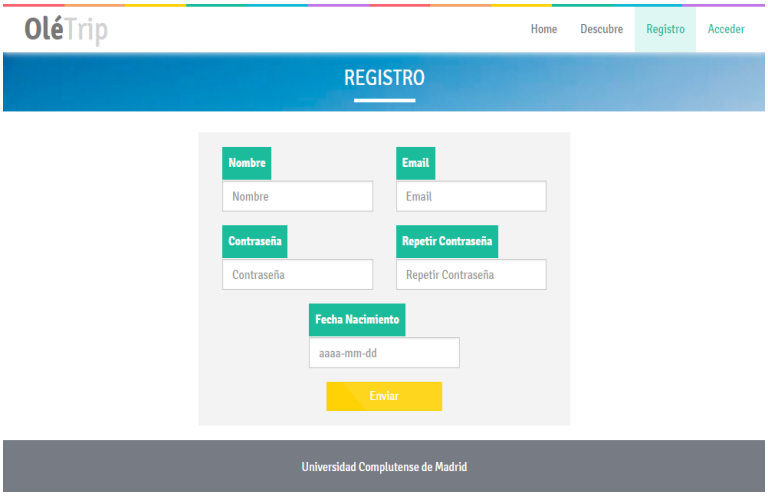
En ella se podrán ver las diferentes actividades que se pueden encontrar en el sistema, estás estarán clasificadas por provincia, y a su vez por categorías. Es decir, en primer lugar, se encuentran las diferentes provincias, en las que se puede seleccionar cual ver, una vez seleccionada, te mostrará las diferentes categorías disponibles, así como una muestra de actividades. Si se selecciona una categoría, mostrará todas las actividades pertenecientes a esa categoría, y la provincia seleccionada con anterioridad. Y si se accede a la actividad, mostrará información relevante de la misma.

Para esta parte, se emplean los servicios, de `/site/city/{provincia}/categories` y `/site/{id_site},` el primero devuelve las categorías existentes en cada provincia, y el segunda la información relativa a un lugar.

5.5 INTEFAZ. PÁGINA DE DESCUBRE

- ***Página de registro***

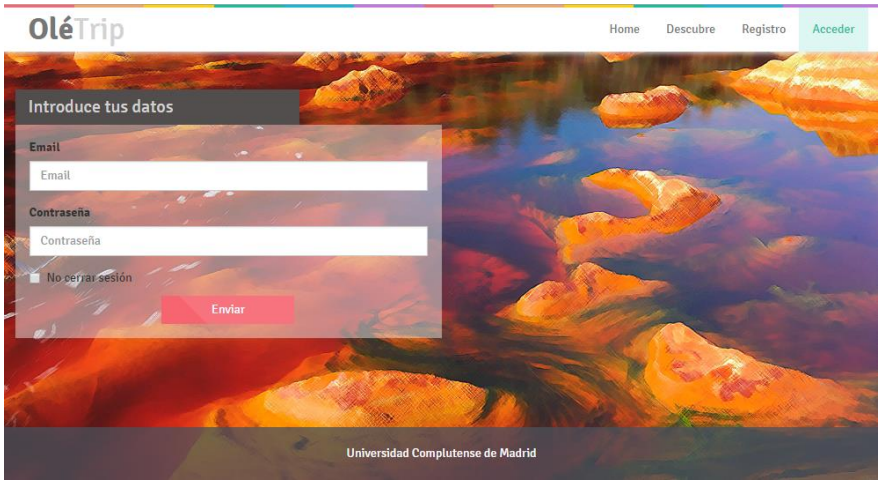
Los usuarios que lo deseen, al acceder a esta página se encontrarán con un formulario de registro, en el cual se requiere: nombre, email, contraseña y fecha de nacimiento, y previamente al darle efectuar el registro se comprueba que los campos no son vacíos, que las contraseñas sean iguales, y que la edad del usuario sea más de 14 años, en el caso de que no se cumpla alguna de las condiciones anteriores, no se efectuará el registro y se le notificará al usuario, pero si por el contrario todo ha ido bien, el usuario será registrado mediante el servicio `/user/signup` y podrá disfrutar de todas las funcionalidades de OléTrip, si el servicio responde con los datos deseados.



5.6 INTERFAZ. REGISTRO

- ***Página de acceso***

Los usuarios registrados podrán acceder a las funcionalidades de la web, una vez que hayan accedido a través de la página de login, en la que tendrán que proporcionar su email, y contraseña, y estos no han de ser vacíos, y para verificar que el usuario está en el sistema, se consulta mediante el método `/user/login`, proporcionándonos en el caso correcto, los datos para controlar la sesión.



5.7 INTERFAZ. PÁGINA DE ACCESO

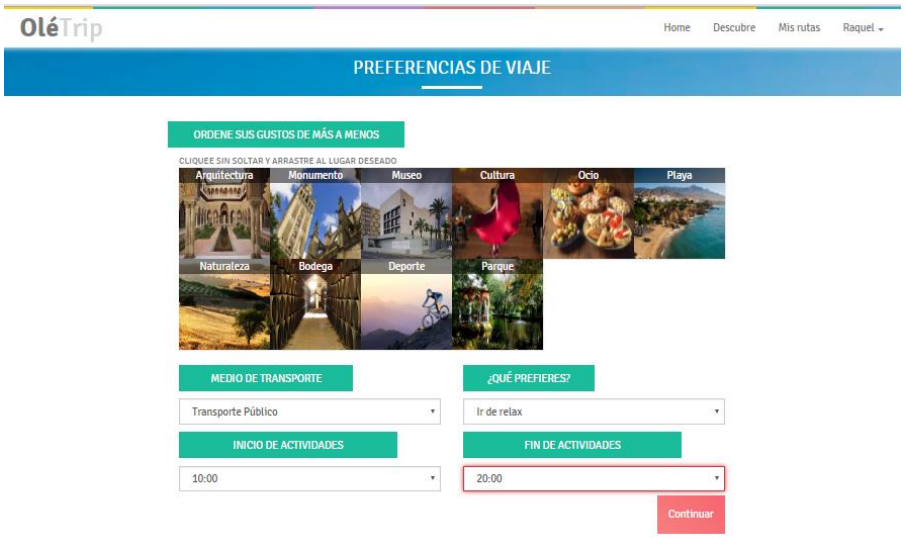
- ***Servicios a registrados***

A parte de los ya comentados con anterioridad, los usuarios pertenecientes a la aplicación podrán acceder a:

- ***Página de preferencias de viaje***

Tras seleccionar, los días y provincias a visitar, para poder conocer más lo que el usuario desea, se le pide que seleccione, las preferencias para ese viaje, ordenándolas de mayor a menor, para ello se le muestran las imágenes y el tipo de actividad, y este ha de arrastrar al lugar deseado, además de proporcionar el medio de transporte, coche o transporte público, el tipo de viaje, ir de relax o ver todo, así como la hora de inicio y fin de las actividades, que podrá escoger entre un rango de horarios.

Tras recolectar las preferencias, se crea el viaje, para ello se realiza una llama al método, `/trip/create`, que nos devolverá el identificador del trip, para posteriores funcionalidades.

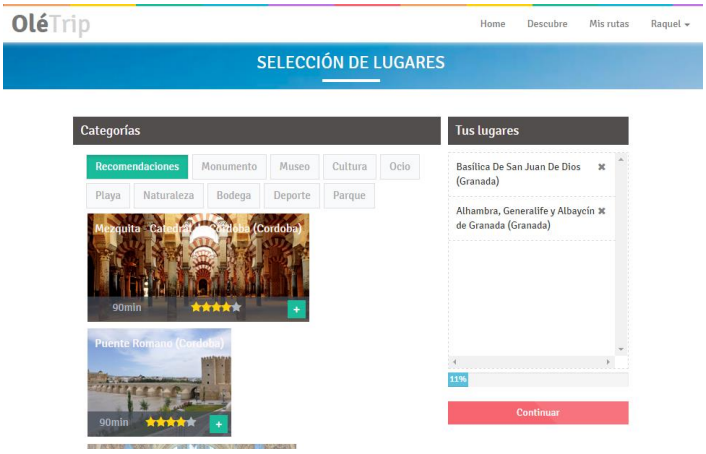


5.8 INTERFAZ. PREFERENCIAS DE VIAJE

- ***Página de selección de lugares***

Aparecen las actividades, ordenadas por las preferencias del usuario, destacándole las recomendadas para él. Para mostrarlas se hacen peticiones a los servicios `/site/city/{provincia}/category/{categoria}` y, `/site/{idSite}`, el primero devuelve las actividades pertenecientes a cada provincia y categoría, y el segundo los datos de una actividad específica.

El usuario, tendrá que añadir las actividades que desee, arrastrándolas al carrito o dándole al botón de añadir, y no se dejará pasar a la creación de la ruta hasta que no se rellene la mitad del tiempo, que se empleará.

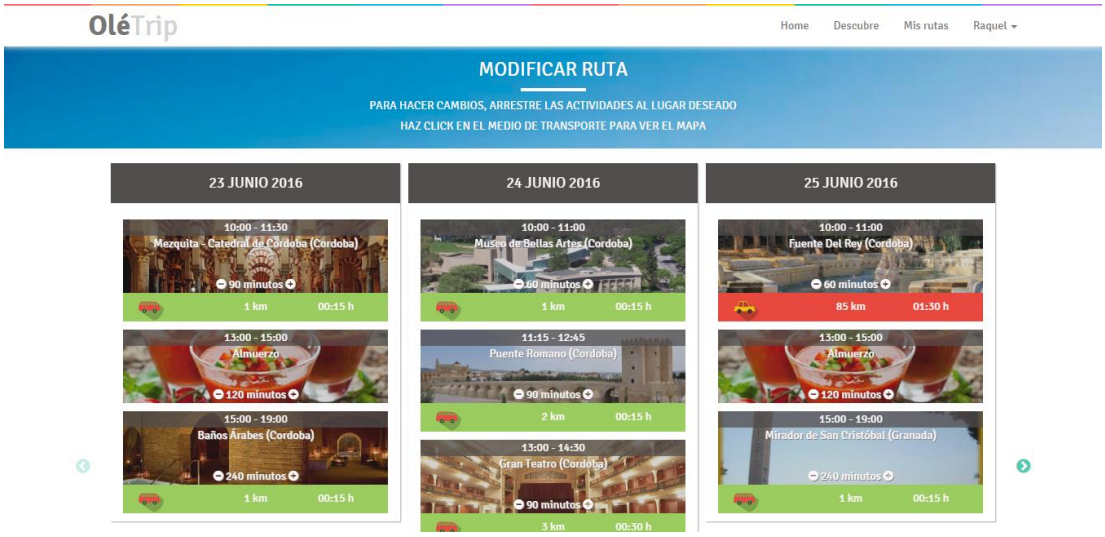


5.9 INTERFAZ. SELECCIÓN DE LUGARES

- ***Página de previsualización y modificación de ruta***

Tras la llamada al servicio `/trip/{id_trip}/rute`, si todo ha ido bien, nos devuelve la ruta, la cual será mostrada, separando cada día, y mostrando en cada uno las actividades idóneas, para ese día, en cada una se verá reflejada la hora de inicio y fin, la duración, y el tiempo y distancia a la siguiente actividad, así como el medio de transporte. Cabe destacar que toda la ruta es alterable, si el usuario lo desea puede cambiar la duración de cada actividad, el día en el que se quiere realizar, y se volverá a recalcular los tiempos entre actividad y actividad. Esto se consigue con elementos 'draggables' y 'dropables', que permiten arrastrar las actividades al lugar deseado.

Cuando el usuario decide que el plan es el idóneo, podrá guardarlo, llamando así internamente al servicio `/trip/{id_trip}/save`, en el que con jquery se va recolectando el orden de las actividades, y la información necesaria de cada una, que estará disponible como atributos en sus elementos HTML, y si es correcto se le redirigirá a la página de previsualizaciones y votaciones.



5.10 INTERFAZ. PREVISUALIZACIÓN Y MODIFICACIÓN DE RUTA

- ***Página de previsualización y votaciones de ruta***

Página similar a la de modificación de ruta, con la diferencia, de que en ella las actividades no se pueden alterar, al no ser que seas el creador, y la actividad no tenga buenas votaciones, que se da la oportunidad de cambiar la actividad por alguna alternativa proporcionada por el servicio `/trip/{id_trip}/activity/{id_act}/alternative`, y modificada por el `/trip/{id_trip}/activity/{id_act}/update`.

En ella también se puede, dar a me gusta y no me gusta de una actividad, y comentar en ella, así como ver la información relativa a cada actividad (lugar, descripción, web, etc.). Sin embargo, no se podrán dar opiniones si el viaje es definitivo.

Esta es la página que se vale de más servicios, ya que en ella se recolecta toda la información necesaria para describir un viaje. Los servicios que se emplean son:

Información general del viaje, es decir, los horarios, invitados, creador, días, etc.

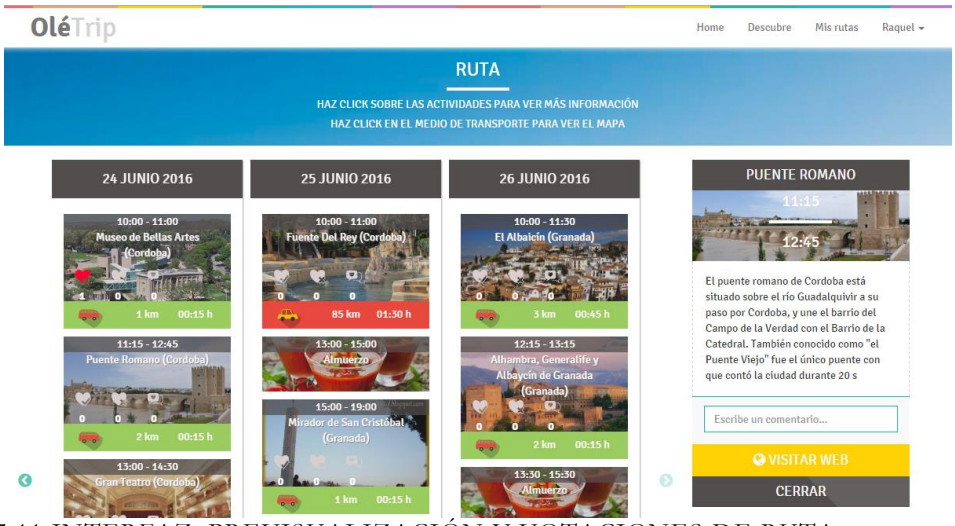
`/trip/{id_trip}`

```
/trip/{id_trip}/owner
/trip/{id_trip}/invited
```

Información específica de cada actividad, así como las opiniones con las que cuenta, y servicios que permiten reflejar opiniones.

```
/trip/{id_trip}/activity/{idA}
/trip/{id_trip}/activity/{idA}/vote
/trip/{id_trip}/activity/{idA}/user/vote
/trip/{id_trip}/activity/{idA}/comments
/trip/{id_trip}/activity/{idA}/likes
/trip/{id_trip}/activity/{idA}/dislikes
```

Además, se permite desde aquí invitar a amigos, llamando así al servicio `/trip/{id_trip}/invite,` siempre y cuando sea el creador de la ruta.



5.11 INTERFAZ. PREVISUALIZACIÓN Y VOTACIONES DE RUTA

- ***Página de perfil***

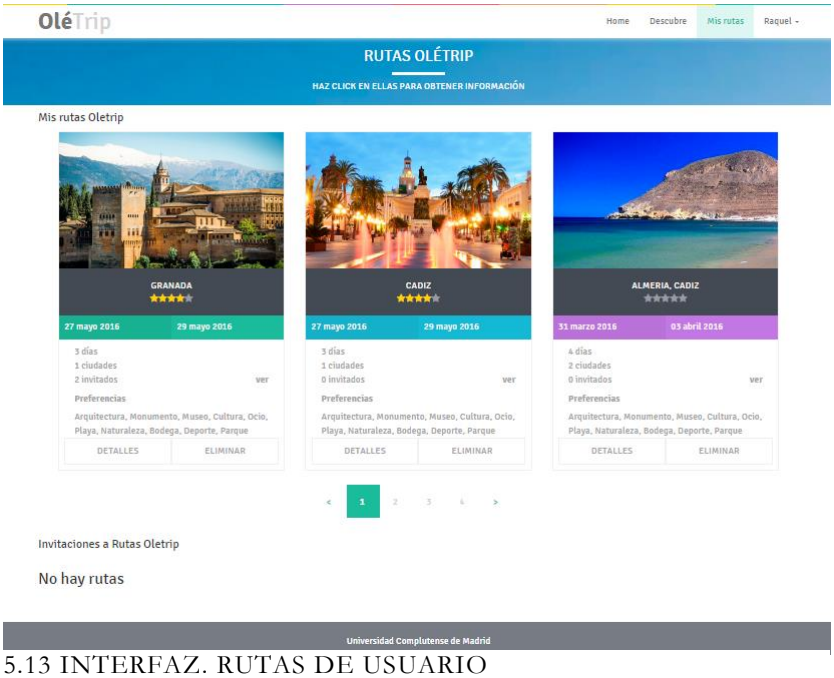
En ella se ve reflejada los datos básicos del usuario, nombre, email y fecha de nacimiento, permitiéndole en cualquier momento modificarlos. Para ello se emplea el servicio `/user/{id_user}` que nos devolverá su información, y `/user/{id_user}/update,` para modificar sus datos.

5.12 INTERFAZ. PERFIL DE USUARIO

- ***Página de rutas***

Cada usuario registrado podrá ver las rutas que ha creado y a las que ha sido invitado, en cada una se podrán ver los invitados, los días, provincias, preferencias, así como acceder a la página de previsualización, eliminar invitados y eliminar una ruta. Éstas dos últimas funciones sólo podrán realizarse si eres el usuario creador.

Para poder visualizar y realizar estas acciones, se han llamado a los servicios `/trip/{id_trip}`,



5.13 INTERFAZ. RUTAS DE USUARIO

`/trip/{id_trip}/user/{id_user}/noinvite` y `/trip/{id_trip}/delete`.

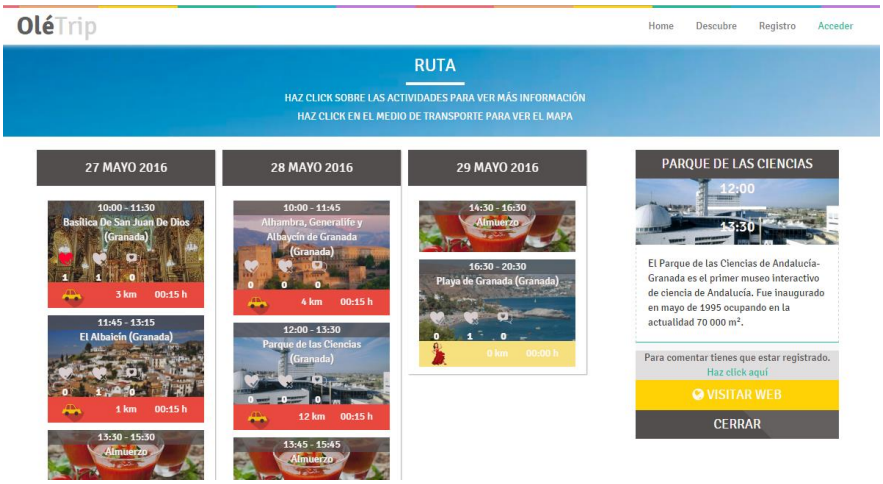
- Servicios a no registrados
  - ***Página de sugerencias***

    Como no está registrado no tiene la opción de crear un viaje personalizado, por ello, se le dan una serie de recomendaciones basadas en las provincias que va a visitar, empleando el servicio `/trip/recommend`, y si este no devuelve ninguna recomendación se incita al usuario a registrarse para poder ayudarlo a encontrar su ruta perfecta.

5.14 INTERFAZ. SUGERENCIA DE RUTAS

- ***Página de previsualización y votaciones (usuario invitado no registrado)***

Tendrá las mismas características que en la de usuarios registrados, con la excepción de que no se podrá realizar ningún comentario, para ello se ha de registrar, como ya se ha comentado.



5.15 INTERFAZ. PREVISUALIZACION Y VOTACIONES (NO REGISTRADO)

5.5. Mecanismo de recomendación

Existen dos mecanismos de recomendación, uno diseñado para usuarios pertenecientes (registrados) a la aplicación, y otro para usuarios no pertenecientes (no registrados).

5.5.1. Usuarios no registrados

Para el de usuarios no pertenecientes al sistema, el algoritmo se basa en un filtro colaborativo, es decir, tras proporcionar las provincias que se desea visitar, el algoritmo comprueba que en la base de datos haya viajes ya creados con tales provincias, si es así, se recomiendan aquellos viajes con mejores valoraciones, es decir, es lo que hacen los sistemas actuales.

5.5.2. Usuarios registrados

Por otra parte, en el otro mecanismo de creación de plan:

Para crear un viaje totalmente personalizado se requiere que el usuario establezca ciertos criterios sobre el viaje realizar, para ello se han de especificar las preferencias mediante el servicio /trip/create, donde los parámetros necesarios para crear un viaje, previamente sin actividades, son:

- Lista de provincias a visitar.
- Fecha inicio del viaje.
- Fecha fin del viaje.
- Lista ordenada de las preferencias del usuario.
- El medio de transporte preferible a utilizar.
- Hora inicio del día para empezar las actividades.
- Hora fin del día para terminar las actividades.

Tras ello, el usuario ha de proporcionar lugares que desea visitar, mediante la interfaz diseñada, se le recomiendan actividades en base a las preferencias seleccionadas. Y estás se han de proporcionar mediante el servicio `/trip/{id}/rute`.

Con los sitios proporcionados la aplicación calculará una ruta completa para los días seleccionados, utilizando el algoritmo voraz mejorada con algoritmo de agrupación K-means. [33]

El algoritmo de K-Means, conocido también como algoritmo de Lloyd, tiene resultado de aglomerar N observaciones en K grupos, estos dependen de la cercanía de las observaciones respecto a los centroides de los mismos. Se llama centroide al punto que define el centro geométrico del grupo dividido, en este caso, vendrá definido por un par de coordenadas de latitud y longitud. Este algoritmo complementa al algoritmo voraz, ya que éste puede generar recorridos erróneos o incompletos, al calcular la ruta mediante el camino mínimo, causando alguna excepción como dejar algún sitio de la provincia sin recorrer por estar más aislado o lejano.

El algoritmo va construyendo el recorrido de un sitio a otro, sin repetir ninguno de ellos. Y a la vez que los recorre, calculará otros factores como

medio de transporte, slots de tiempo empleado para cada actividad y transporte, reservar hora para la comida, etc.

Para el cálculo de distancias, medio de transporte y tiempo empleado en dicho desplazamiento, se realizan distintas peticiones al servicio de Google Maps Matrix Distance API[6]. Este devolverá error cuando la petición realizada con el medio de transporte preferente del usuario no exista, en este caso se calculará dicha ruta con otros tipos de transportes alternativos.

Un ejemplo de petición sería el siguiente:

```
https://maps.googleapis.com/maps/api/distancematrix/json?origins=36.5968944475872,-6.23282950372602&destinations=36.7760949432912,-6.35319414166177|36.4577525225489,-6.20363489980176&key={api-key}
```

También cabe destacar, que se ha implementado también un pequeño algoritmo para recomendar sitios alternativos a la hora de sustituir una actividad de un viaje por otra, ya sea porque a los integrantes no les hayan gustado o por otros motivos, este mecanismo será comentado posteriormente en el apartado **0** del Mecanismo de retroalimentación.

5.6. Mecanismo de creación de ruta

- **Fase I – Inicialización**

Siendo $\beta = \{\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, \mu_8, \mu_9, \ldots, \mu_n\}$ el conjunto de actividades seleccionadas para realizar el viaje, éste se subdividirá en ***n*** conjuntos, correspondiendo a ***n*** al número total de provincias a visitar.

Dicha división viene dada por:

```
Inicio
      α = array[n]
      provincias = getTritProvincias(id_trip)
      Para i = 0 hasta n hacer
            aux = array[]
            cont = 0
```

[6] https://developers.google.com/maps/documentation/distance-matrix/intro?hl=es

```
            Para j = 0 hasta tamaño de β hacer
                  Leer β(j).ciudad
                  Si β(j).ciudad == provincias(i)
                        aux[cont++] = β(j)
                  FinSi
            Fin_Para
            α [i] = aux
      Fin_Para
Fin
```

Suponiendo que $\beta = \{\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, \mu_8, \mu_9\}$, y n es igual a 2, se obtendría:

$$\text{Provincia } \alpha_1 = \{\mu_3, \mu_4, \mu_5\}$$
$$\text{Provincia } \alpha_2 = \{\mu_1, \mu_2, \mu_6, \mu_7, \mu_8, \mu_9\}$$

Con esta subdivisión se procede a la obtención de clústeres con k-means, teniendo en cuenta que siempre se empezará por la provincia que contenga más actividades.

- **Fase II – algoritmo K-Means**

Como ya se ha comentado, este algoritmo se centra en la creación de los distintos clústeres(φ), es decir, conjuntos de actividades en una determinada provincia, respecto a un punto más cercano a la media(centroide).

Los Centroides(C) a crear para cada provincia se establecen siguiendo la siguiente relación:

$$C = (N/4) + 1$$

Siendo N el número de actividades.

Siguiendo el ejemplo anterior de la subdivisión de β, para la provincia α_2 se obtendrán 2 centroides, y para la provincia α_1 se obtendrá 1 centroide. Estos se calculan en base a los mínimos y máximos del conjunto de coordenadas (latitud (ϕ), longitud (λ)) pertenecientes a cada α (provincia).

$$\text{Centroide}(C_n) = \text{random}[\ \text{mínimo}(\ \alpha_x\varphi[]\ ),\ \text{máximo}(\ \alpha_x\varphi[]\ )\ ]$$

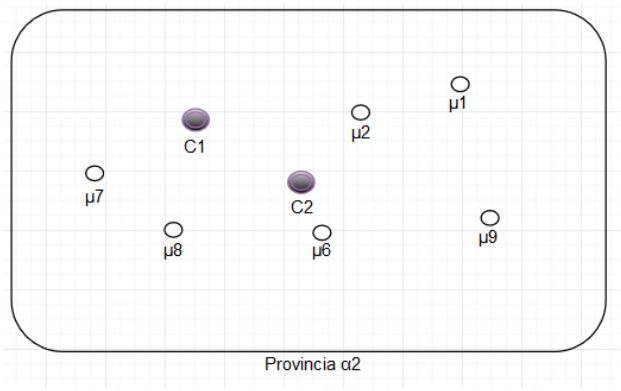
Cabe destacar, que la posición del centroide se irá recalculando según los sitios añadidos a cada clúster. Una vez que se obtienen los clústeres, estos se organizarán por tamaño, otorgando prioridad al clúster mayor sobre el resto.

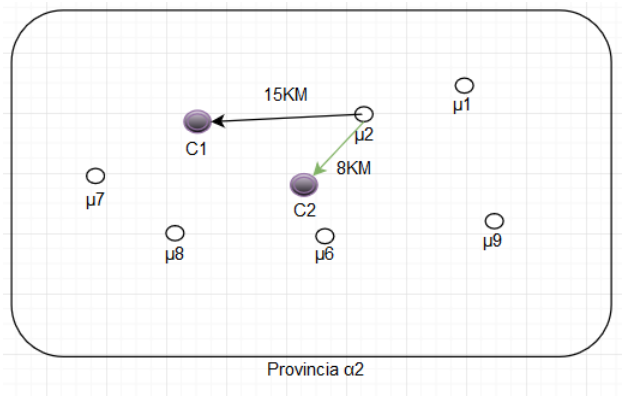
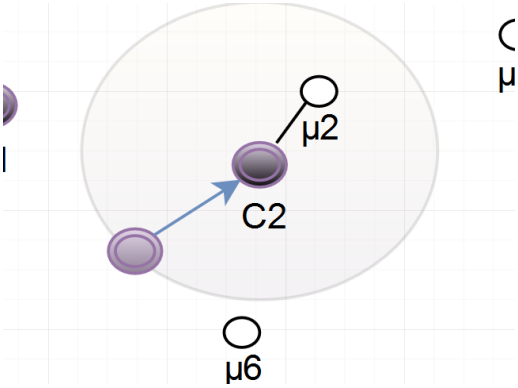
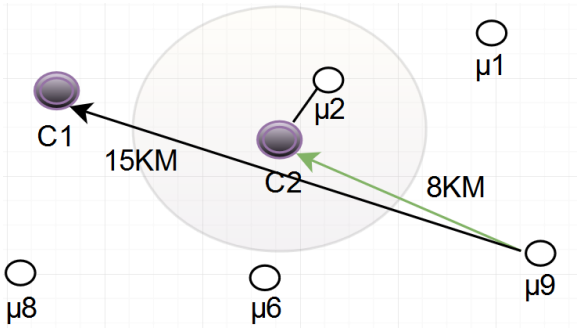
Por otra parte, para el cálculo de distancias (D), se utiliza la siguiente fórmula:

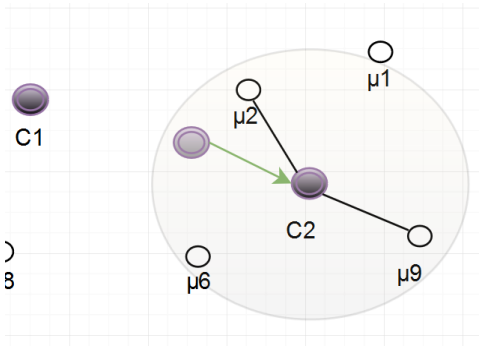
$$D = \sqrt[2]{(\varphi 1 - \varphi 2)^2 + (\lambda 1 - \lambda 2)^2}$$

Al tratarse de una representación cartesiana, dicha distancia se multiplica por 111, que es una aproximación sencilla obtenida mediante la relación entre kilómetros con cada grado sexagesimal. Esto dará un resultado aproximado en kilómetros.

Con ello, y aplicado al ejemplo anterior, el algoritmo de obtención de clústeres se puede ver como:

Imagen y descripción	Sitios no procesados	Clústeres formados
 **Paso 1:** la provincia α2 tiene 6 sitios, por lo tanto generarán 2 centroides aleatoriamente, según la fórmula indicada.	μ2, μ9, μ6, μ1, μ7, μ8	$\varphi_1 = \{\}$ $\varphi_2 = \{\}$

 **Paso 2:** Se van procesando los sitios que aún no se han explorado, calculando la proximidad a los centroides, y se asociará con el más cercano.	μ9, μ6, μ1, μ7, μ8	$\varphi_1 = \{\}$ $\varphi_2 = \{\mu 2\}$
 **Paso 3:** Se recalcula la posición de los centroides según los sitios añadidos	μ9, μ6, μ1, μ7, μ8	$\varphi_1 = \{\}$ $\varphi_2 = \{\mu 2\}$
 Se repite el **paso 2** con el siguiente sitio	μ6, μ1, μ7, μ8	$\varphi_1 = \{\}$ $\varphi_2 = \{\mu 2, \mu 9\}$

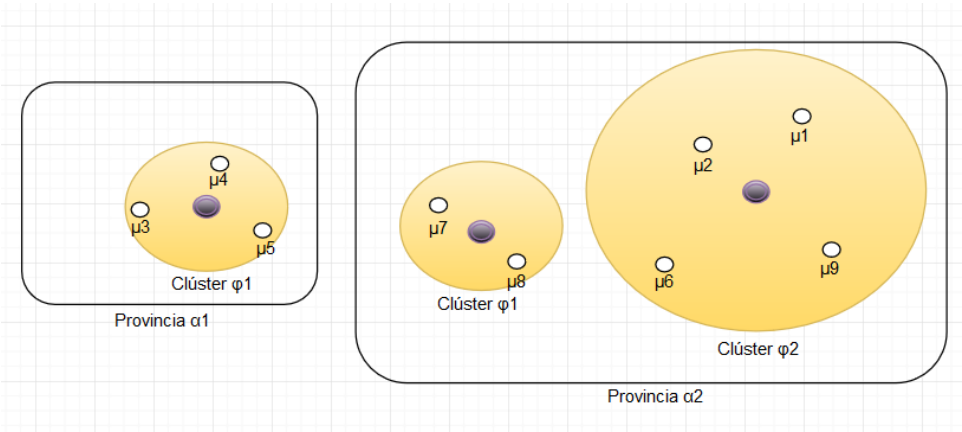
 Se repite el **paso 3** con los datos actuales, y se repetirán los pasos 2 y 3, así sucecivamente hasta terminar de consumir todos los sitios	μ6, μ1, μ7, μ8	$\varphi_1 = \{\}$ $\varphi_2 = \{\mu 2, \mu 9\}$

6 K-MEANS

Aplicado al ejemplo anterior se obtendría:

α_2 = {
    φ_1 = {μ_2, μ_1, μ_9, μ_6}
    φ_2 = {μ_7, μ_8}
}
α_1 = {
    φ_2 = {μ_5, μ_4, μ_3}
}

Y tras la obtención de los clústeres, para obtener la ruta óptima, se emplearía un algoritmo voraz en cada uno, este se explicará a continuación.



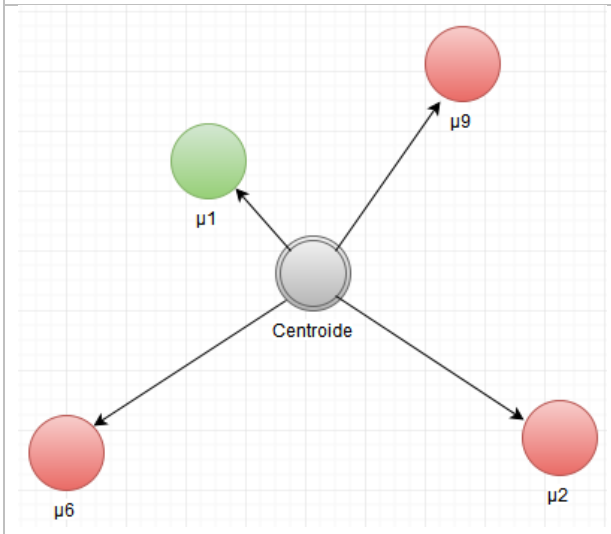
5.16 K-MEANS. CLÚSTER

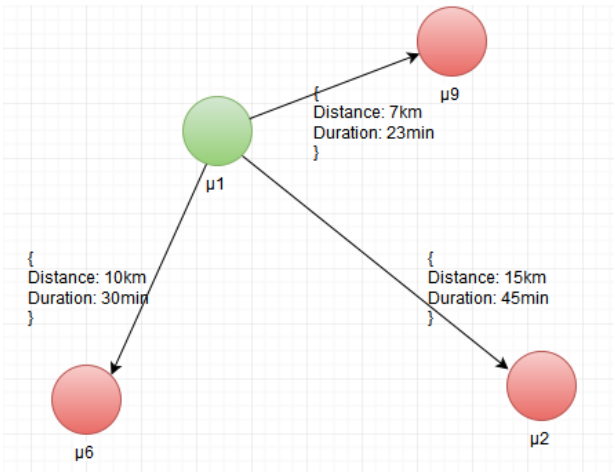
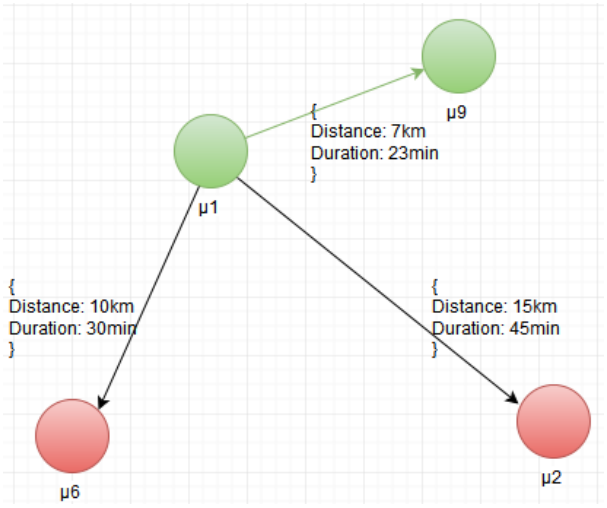
- **Fase III – algoritmo voraz**

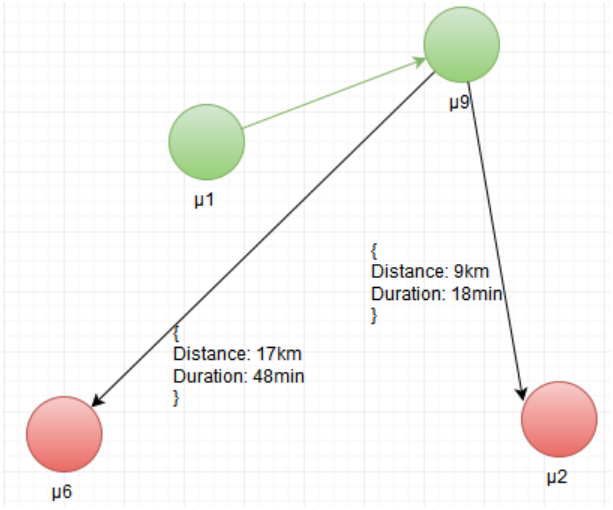
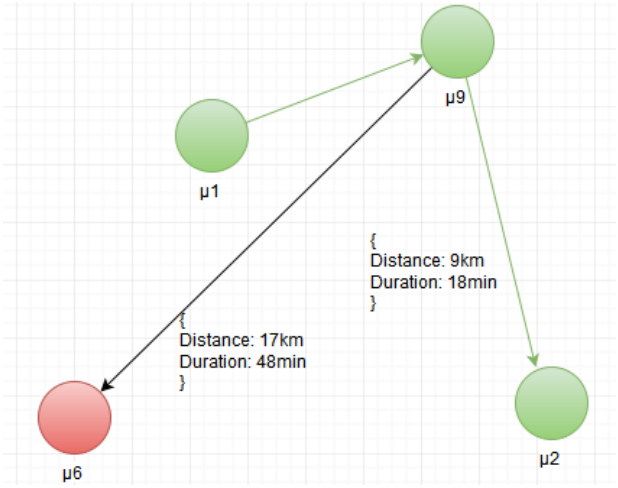
Para llevar a cabo este algoritmo se han empleado peticiones de distancia y duración a Google Maps Matrix API.

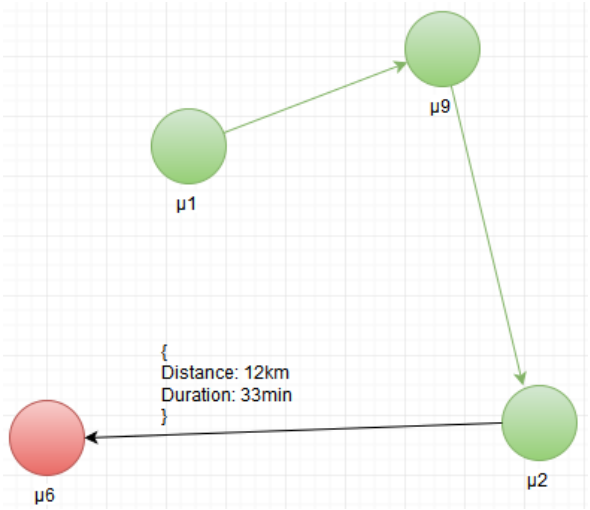
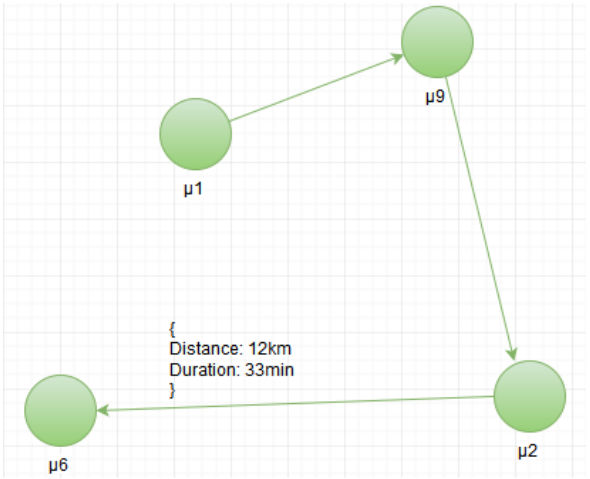
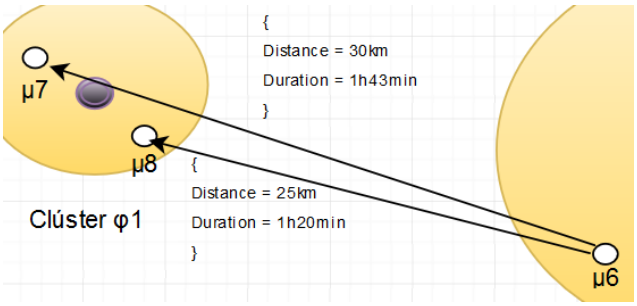
Como ya se ha comentado, se aplicará el algoritmo en cada una de las provincias subdivida en clústeres, este irá creando la ruta por días según la distancia mínima de recorrido. En los cálculos, también influirá la duración de transporte, duración de la visita y tiempo reservado para la comida, si existen actividades que no han sido capaz de añadir al plan, por diversos motivos como falta de tiempo, se informará también en la respuesta de la creación del plan.

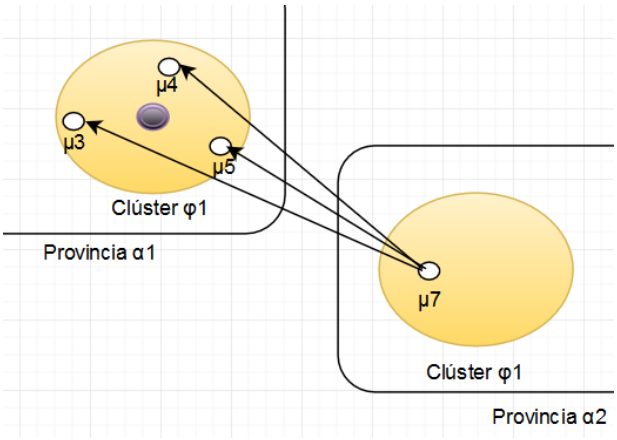
A continuación, se verá la aplicación del mismo, siguiendo el ejemplo anterior, empezando por la provincia 2.

Imagen y descripción	***Sitios no visitados***	***Recorrido actual***
 **Paso 1**: En la primera iteración, el inicio no está definido, por tanto, se define el inicio como el punto más cercano al centroide	μ2, μ9, μ6	μ1

 **Paso 2:** Se realiza una petición a Google Maps Matrix API (origen= μ1 , destino={μ2, μ6, μ9}) a los sitios no visitados para determinar la cercanía al punto de origen.	μ2, μ6, μ9	μ1
 **Paso 3**: Se elige el sitio con valores mínimos, dichos valores se guardan, y posteriormente, se genera el plan completo por días.	μ2, μ6	μ1, μ9

 **Paso 4**: Desde el sitio actual realizamos el paso 2 a los otros dos sitios restantes	μ2, μ6	μ1, μ9
 **Paso 5**: Se escoge el mínimo guardando los valores calculados	μ6	μ1, μ9, μ2

 Se repite el **paso 2**.	μ6	μ1, μ9, μ2
 Se repite el **paso 3** y en este caso, se termina de recorrer el clúster.	-	μ1, μ9, μ2, μ6
 Posteriormente se busca el sitio más cercano al próximo clúster, desde el ultimo sitio añadido, y	-	-

se repiten los pasos 2, 3 y 4 y así sucesivamente hasta recorrer todos los clústeres		
 Una vez que el camino este definido para una provincia, se procede a la siguiente provincia, desde el ultimo sitio añadido al sitio más cercano del primer clúster de dicha provincia.	-	-

Una vez procesado el algoritmo se obtiene el camino mínimo de la ruta y los tiempos empleados en los transportes, elaborando así, el plan completo por tiempo, acorde a los días disponible y tiempo disponible en cada día, para ello se tiene en cuenta la hora inicio y fin fijado por el usuario y los días naturales que durará el viaje. Siempre se calculará todo el camino completo hasta que no quepan actividades, si da este caso, las actividades sobrantes se informarán en la misma respuesta.

Un posible resultado final, sin reflejar los tiempos sería el siguiente:

```
Plan = [
     Día 1 = {μ1 -> μ9 -> μ2 -> comida -> μ6 -> μ8 -> μ7},
     Día 2 = {μ5 -> μ4 -> comida -> μ3}
]
```

En resumen, el mecanismo de recomendación, se basa en la agrupación de lugares cercanos, ordenados en base a la cercanía y el tiempo de transporte, y

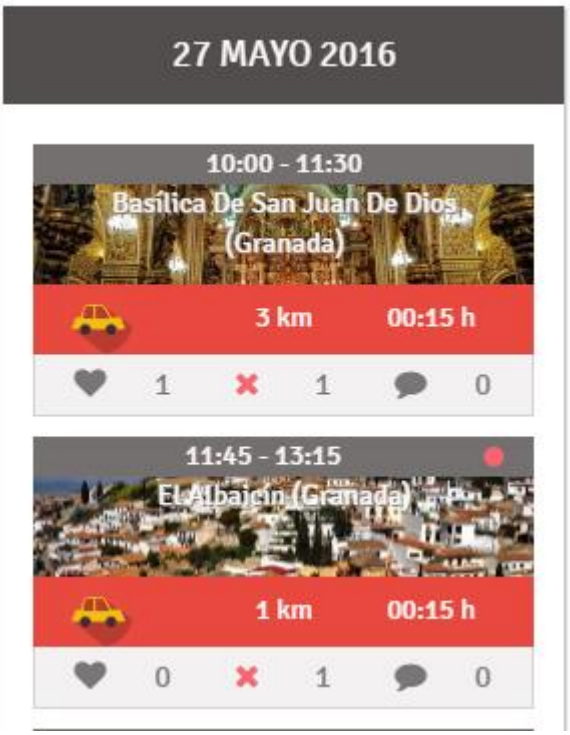
siempre en base a un punto de referencia. Y finalmente, se ve reflejado como una lista ordenada y detallada.

5.7. Mecanismo de retroalimentación

Al tratarse de casos de viajes grupales, los distintos métodos de retroalimentación de opiniones e información sobre el viaje son vitales para generar un viaje que sea satisfactorio para todos en su totalidad. Para ello se han implementado distintos mecanismos para perfeccionar y personalizar aún más el viaje elaborado por los algoritmos.

a) ***Votación de actividades***

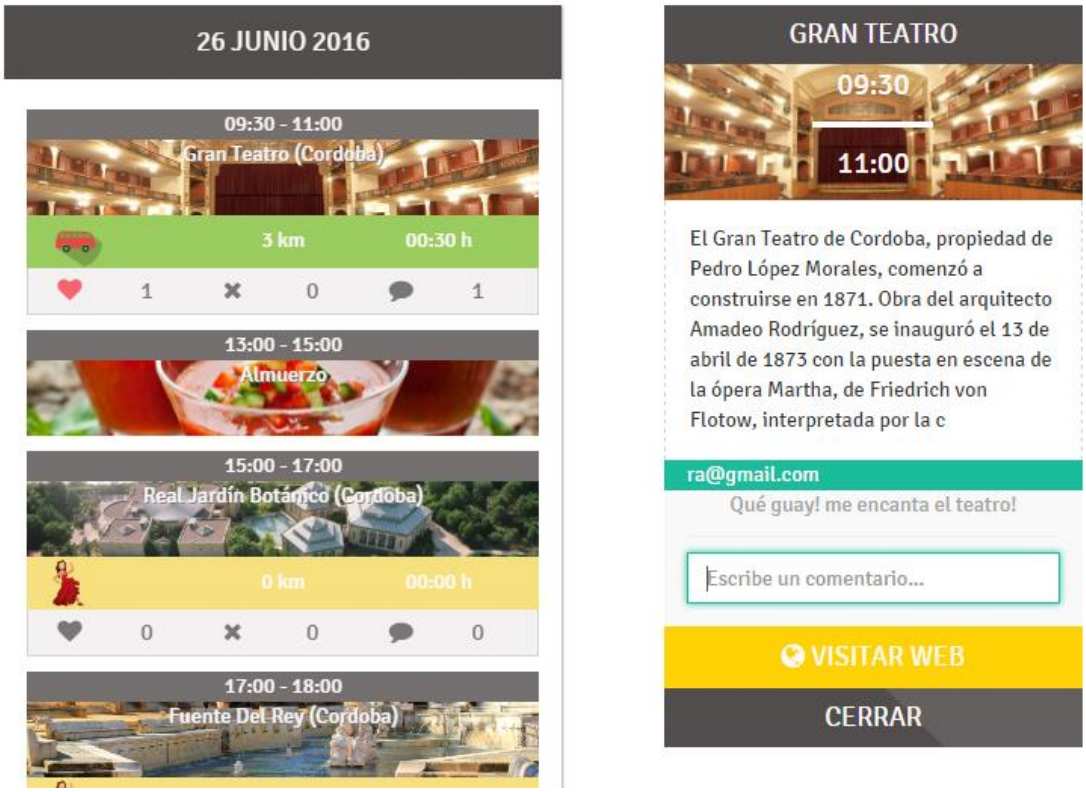
Para esta funcionalidad, se ha creado un sistema de *likes/dislikes*, el cual es muy utilizado en muchas redes sociales como Facebook para conocer brevemente la opinión de los usuarios, en éste caso se le proporciona al planificador del viaje un breve idea de si están gustando o no las distintas actividades.



5.17 VOTACIÓN DE ACTIVIDADES

b) ***Comentario personal sobre actividad***

El sistema de votaciones no es lo suficientemente explícito como representación general de las opiniones personales, por lo que se implementó un sistema de comentarios. Los comentarios son para cada actividad, permitiendo así, tanto a los usuarios que participan en el viaje como al creador debatir sobre las distintas ideas o dudas que vayan sugiriendo.

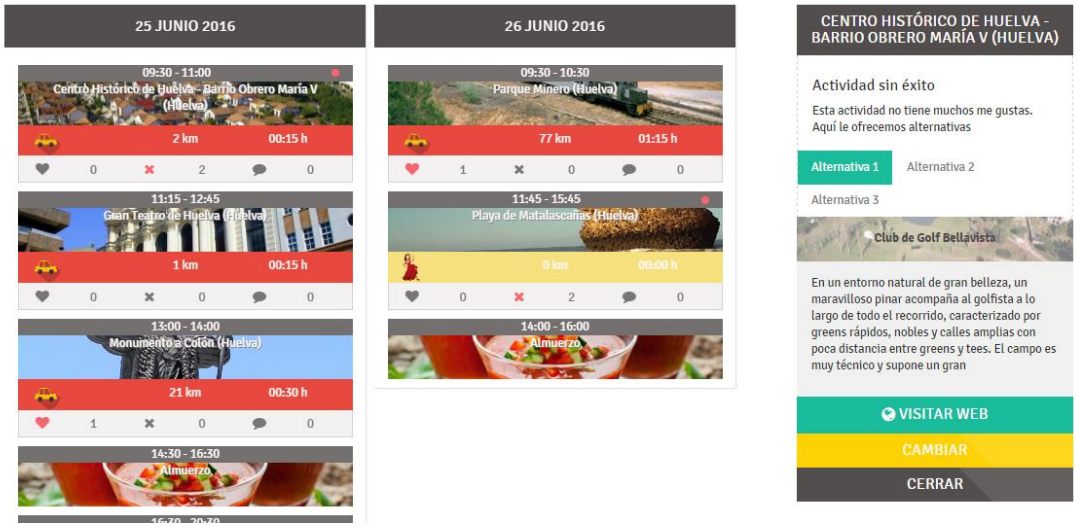


5.18 COMENTARIOS DE ACTIVIDADES

c) ***Reemplazar actividades de un viaje***

Una vez que el creador sepa las distintas valoraciones de los usuarios del viaje a realizar, se puede optar por reemplazar ciertas actividades que han sido votadas negativamente o comentado negativamente por los usuarios. El creador puede lanzar una petición de alternativa, de la cual la API retornará varias alternativas que serán calculadas según los siguientes criterios:

- Solo son seleccionado los sitios de la misma provincia del sitio a sustituir.
- De los seleccionados se descarta aquellos que están más lejos de un radio de 10 kilómetros del sitio a sustituir.
- De los restantes se ordena según las preferencias de viajes elegida por el creador en su momento.



5.19 REEMPLAZO DE ACTIVIDADES

Definitivamente, Olétrip es un sistema de recomendación tanto individual como grupal, que no sólo ofrece información de lugares, sino que además propone una ruta óptima para las actividades deseadas, pudiendo ser modificada por el usuario las veces que quiera, siempre y cuando no lo dé a guardar. Además, permite poder compartir cada ruta, con tus amigos, permitiendo que estos colaboren dando retroalimentación, y en base a ello, poder cambiar actividades posteriormente.

También cabe destacar, nuevamente, que, debido a su estructura modular, puede ser empleada con otros sistemas, y aplicaciones, pudiendo surgir nuevas formas de uso.

6. Evaluación con usuarios

En este capítulo se describirá el proceso seguido para realizar la evaluación con usuarios, y lo que ello ha supuesto para la aplicación.

6.1. Objetivos de la evaluación

Para asegurar el buen nivel de usabilidad y aceptabilidad de la página web creada, se ha llevado a cabo una evaluación de usabilidad con usuarios.

Esta evaluación tiene objetivo poner a prueba la aplicación web con distintos perfiles de usuarios para ver si está acorde o no con los requerimientos y necesidades de cada uno, y estudiar las posibles mejoras.

6.2. Participantes

Se han elegido una media de 10 personas de distintos perfiles para realizar las evaluaciones, estas fueron de distintos rangos de edad, sexo, conocimientos sobre uso de tecnologías webs, estudios y profesión.

Los participantes que participaron en la evaluación fueron principalmente familiares y amigos cercanos de los autores del trabajo fin de grado.

6.3. Encuesta inicial

En un principio, se les ha preguntado por los datos básicos como son: edad y sexo. Aparte de esto, se le han formulado preguntas sobre por la experiencia previa a aplicaciones similares y de su experiencia en navegación web.

Al tratarse de un sistema multiplataforma, esto conlleva que la aplicación podrá ser utilizada en cualquier lugar, por ejemplo: desde el móvil/Tablet

en el bus de vuelta a casa, desde el móvil en una cena de amigos, en frente del ordenador con los familiares, etc… Por lo que se decidió añadir el lugar de la evaluación también como una pregunta del cuestionario inicial.

6.4. Moderador

Cada uno de los integrantes de este trabajo fin de grado ha asumido el rol de moderador, quienes han realizado un breve resumen del propósito de la aplicación web al evaluador. Así como, resolver dudas o confusiones que han ido sugiriendo a lo largo de la evaluación. Los moderadores han tenido que ir anotando los diversos hitos sobre los errores, malinterpretaciones, gestos inadecuados sobre el uso de la web.

6.5. Desarrollo de la evaluación

Cada usuario ha realizado la evaluación acompañado de uno de los integrantes del proyecto, que como se ha explicado, ha asumido el rol del moderador, el usuario tuvo que rellenar un cuestionario inicial y realizar estas tareas definidas:

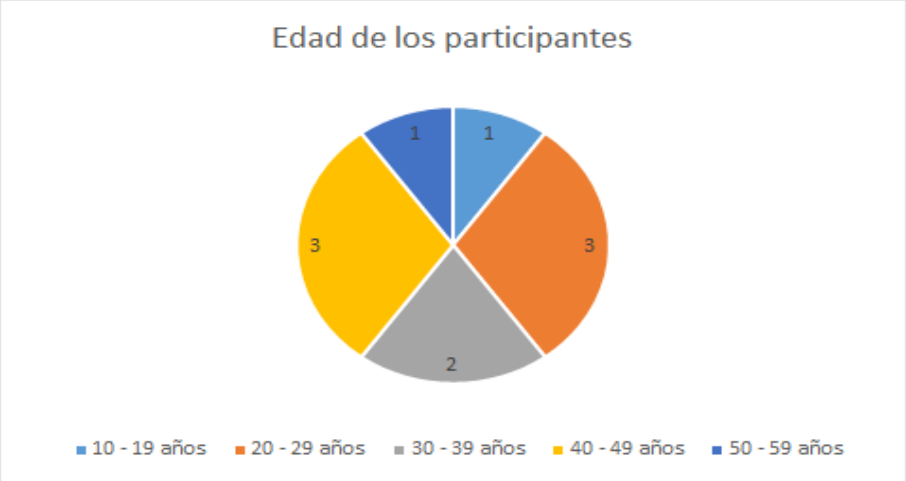
- Registrarse en la página
- Hacer logout
- Hacer login
- Explorar rutas existentes
- Elaborar un viaje personalizado a sus gustos
- Invitar al modelador y a un amigo a este viaje
- Dar "Me gusta" a alguna actividad
- Dar "No me gusta" a alguna actividad.
- Sustituir una actividad que fue votada negativamente por el moderador
- Cerrar las votaciones y marcar el viaje como definitivo.

6.6. Encuesta final

Una vez acabada la evaluación se ha tenido que completar otro un cuestionario sobre el uso general de la aplicación y varias preguntas específicas sobre la fase de creación del viaje, las preguntas de este cuestionario se ponderaron de 1 al 5 (1=muy difícil;5=muy fácil), del cual se han obtenido resultados de medición cuantitativa del proceso. Dicho cuestionario fue acompañado de una pregunta abierta sobre las posibles mejoras de la aplicación.

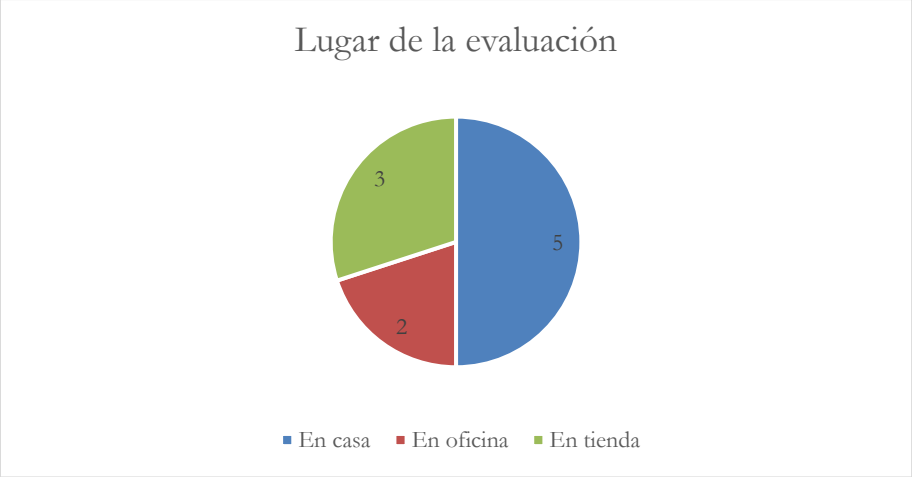
6.7. Resultados obtenidos

- ***Información obtenida del cuestionario inicial***

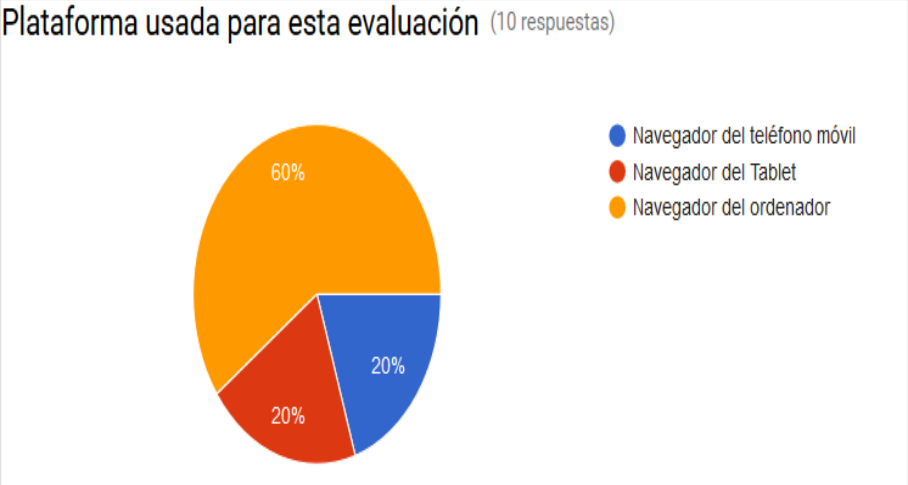


6.1 EVALUACIÓN CON USUARIOS. EDADES

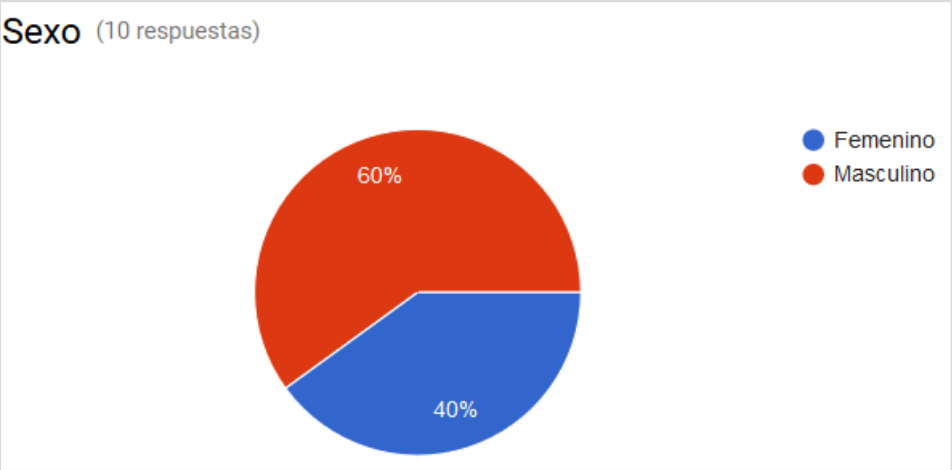
6.2 EVALUACIÓN CON USUARIOS. NAVEGACIÓN



6.3 EVALUACIÓN CON USUARIOS. UBICACIÓN

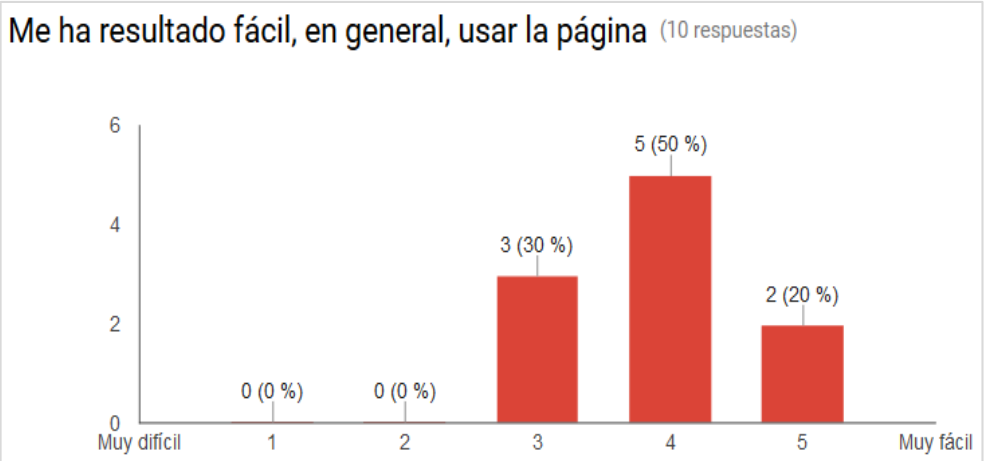


6.4 EVALUACIÓN CON USUARIOS. PLATAFORMA

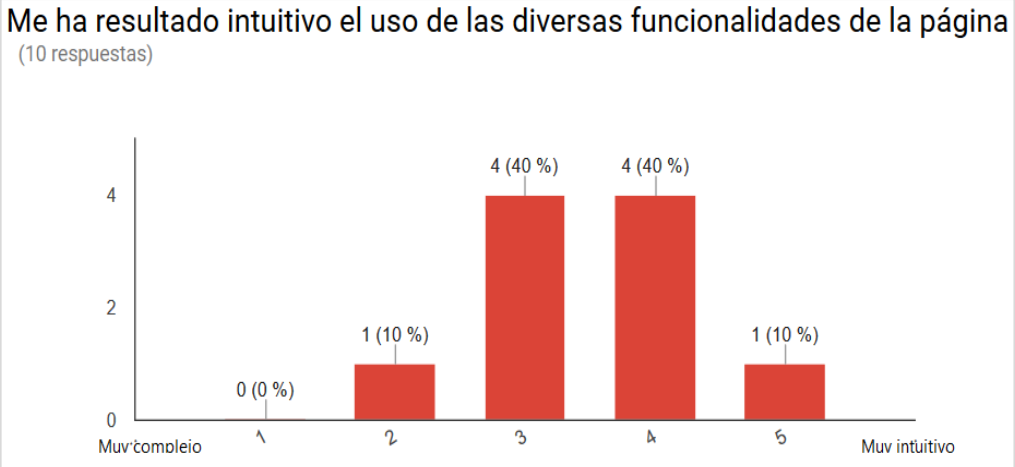


6.5 EVALUACIÓN CON USUARIOS. SEXO

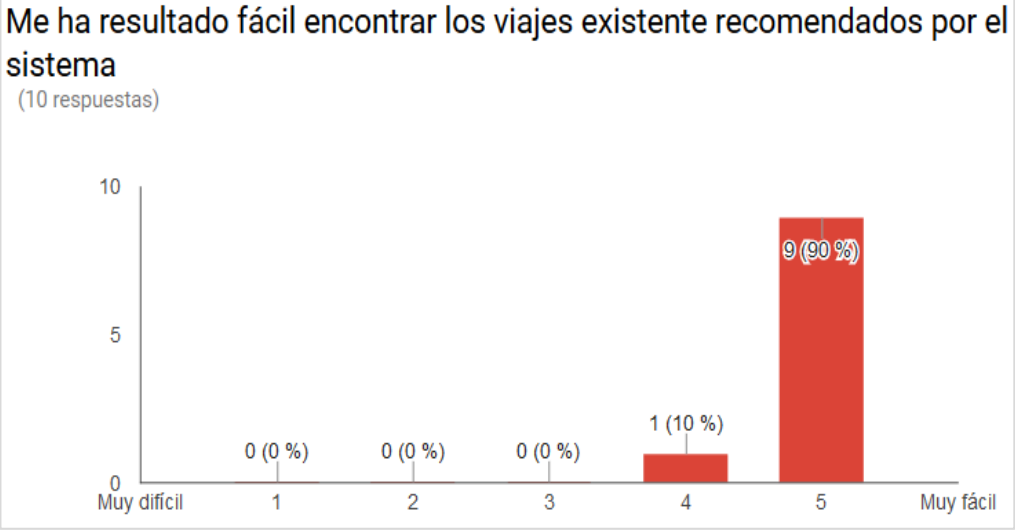
- ***Repuestas del cuestionario final***



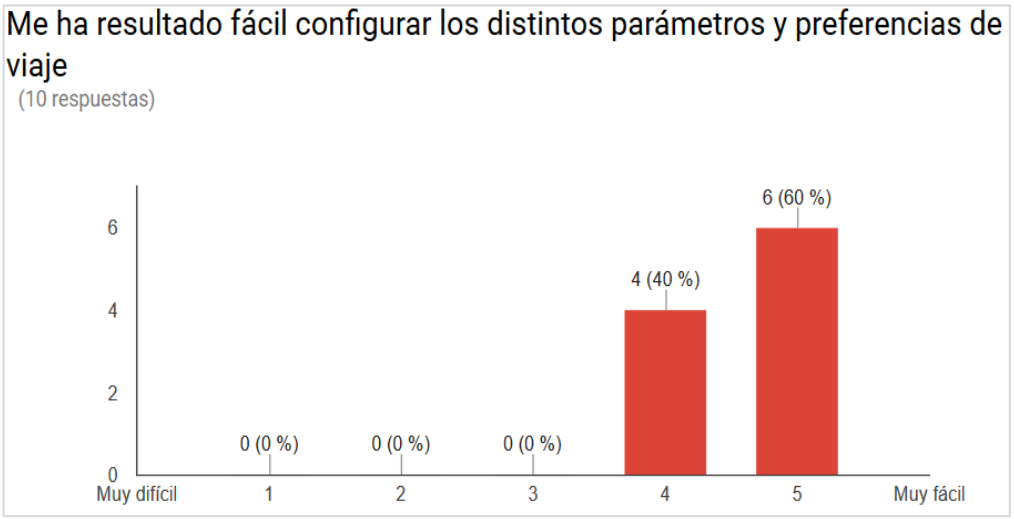
6.6 EVALUACION DE USUARIOS. USO DE LA WEB



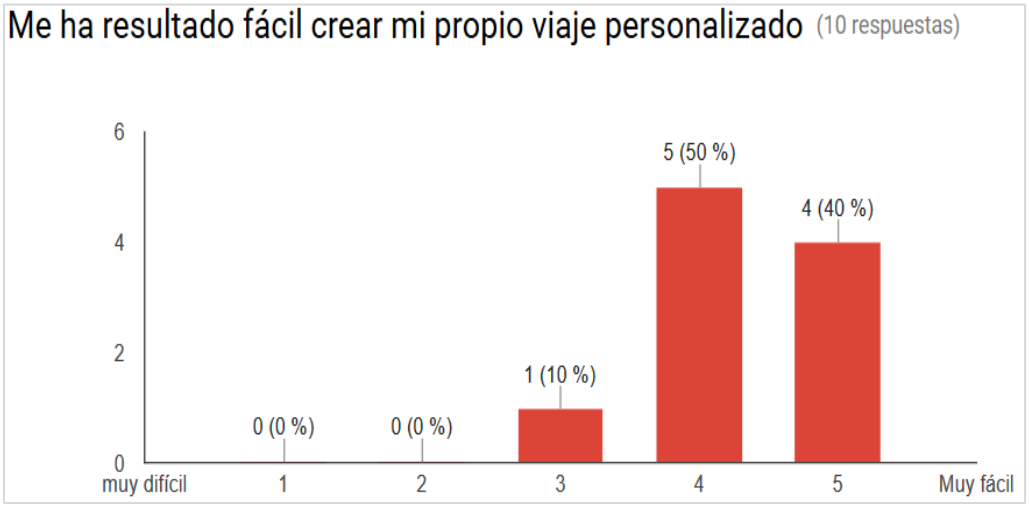
6.8 EVALUACIÓN CON USUARIOS. FUNCIONALIDADES



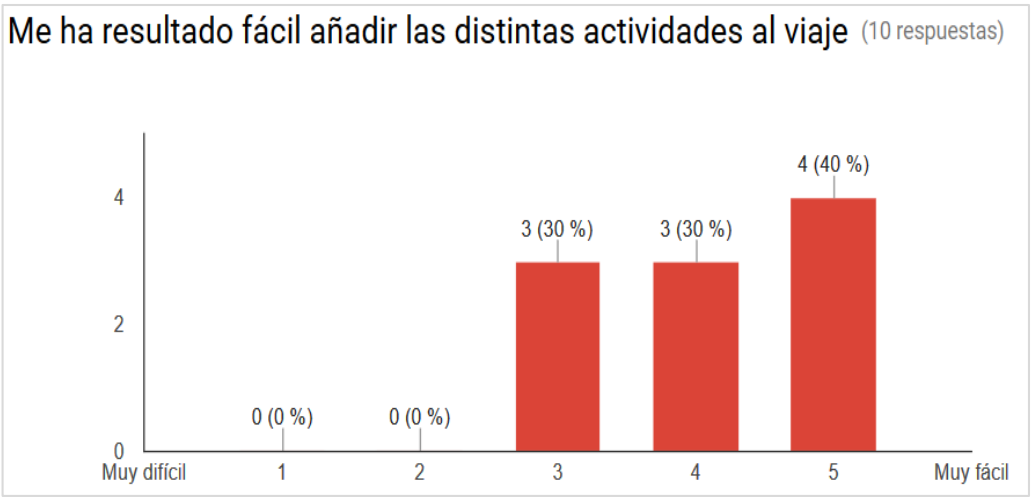
6.7 EVALUACIÓN CON USUARIOS. VIAJES EXISTENTES



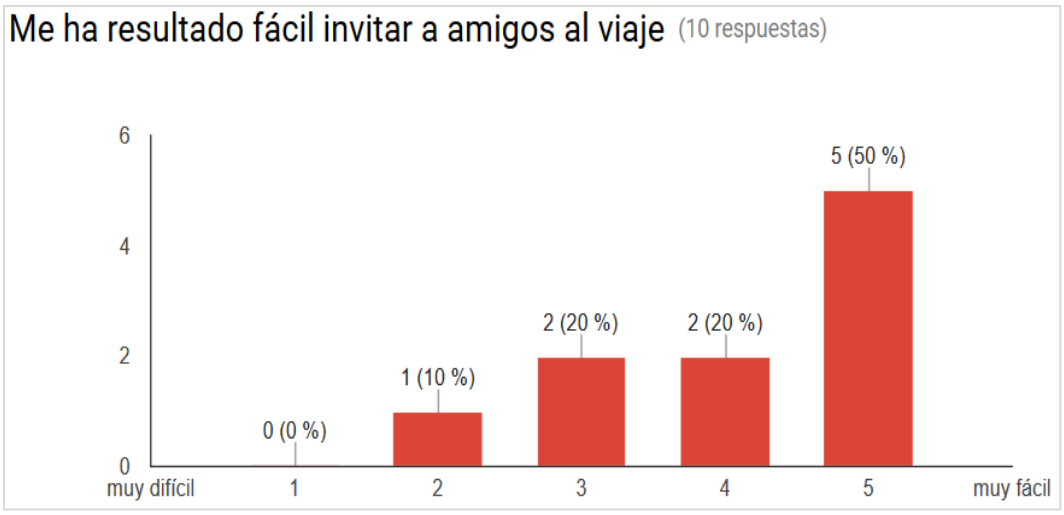
6.9 EVALUACIÓN CON USUARIOS. PREFERENCIAS



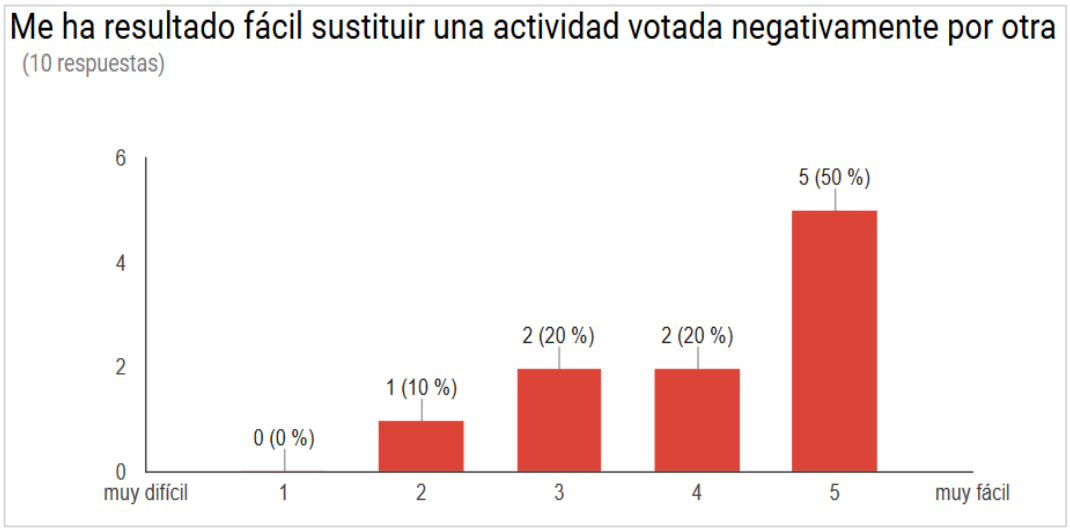
6.10 EVALUACIÓN CON USUARIOS. CREACIÓN DE UN VIAJE



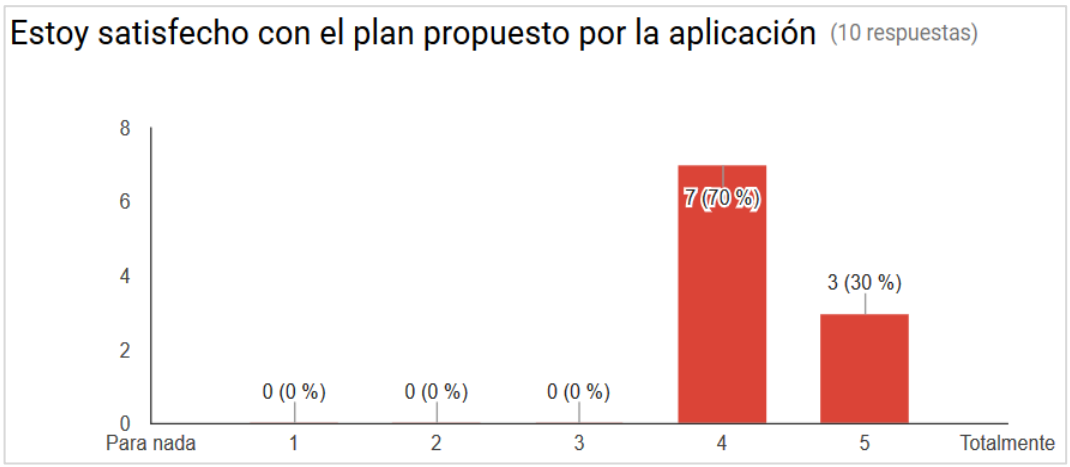
6.11 EVALUACIÓN CON USUARIOS. AÑADIR ACTIVIDADES



6.14 EVALUACIÓN CON USUARIOS.INVITAR AMIGOS



6.13 EVALUACIÓN CON USUARIOS. SUSTITUIR ACTIVIDADES



6.12 EVALUACIÓN CON USUARIOS. PLAN PROPUESTO

6.8. Mejoras concretas a realizar

De los datos extraídos en las diez evaluaciones explicadas anteriormente, se ha podido detectar una serie de problemas en cuanto usabilidad. Son los siguientes:

- No se puede ver la ruta general en el mapa.
- No se pueden ver detalles de las actividades en el proceso de selección de actividades.
- No son visibles los botones de 'Me gusta', 'No me gusta' y 'Comentar'
- Dudas provocadas por 'Arrastre aquí las actividades...' en el proceso de selección de actividades.
- No es posible hacer invitaciones de una sola vez a varios usuarios.
- Mala visibilidad de las funcionalidades de invitar, guardar, finalizar, etc.

Además, los usuarios han sugerido alguna mejora, o funcionalidad que se podría incorporar en un futuro:

- Posibilidad de descargar la planificación del viaje en pdf.
- Incluir hoteles y/o restaurantes en la planificación.

Por tanto, se ha decidido qué problemas serán tratados y cuáles no, y para aquellos que van a ser tratados, se han propuesto una serie de soluciones, y priorizadas para llevarlas a cabo de la manera más óptima y organizada posible.

- Prioridad alta
    - Visibilidad de los botones

      Para que los usuarios encontrasen más rápido los botones de funcionalidad (invitar, salir, guardar, etc.), se han redistribuido, colocándose en la parte derecha de la pantalla, por lo tanto, siempre serán visibles.

- Detalles de los lugares

  Para solucionarlo, se ha añadido un botón con icono de ojo colocado en la esquina inferior derecha de cada actividad. Cuando el usuario pulsa sobre este icono, aparece más detalles sobre dicha actividad. De esta manera, el usuario puede obtener suficiente información de las actividades antes de añadirlos en la planificación.

- Prioridad media
  - Mapa general

    Para solucionar el problema relacionado con la ruta general en el mapa, se ha añadido un botón 'Ver mapa' en el menú situado a la derecha de la planificación, una vez que el usuario termine el proceso de selección de las actividades, puede pulsar sobre este botón para visualizar en el mapa todas las actividades seleccionadas de la ruta.

  - Iconos de retroalimentación

    Se han cambiado los iconos representativos por iconos más comunes e intuitivos. Además, se han cambiado de lugar, pasando de estar debajo del título y con fondo de imagen, a la parte inferior, y con fondo sólido, para poder destacar más.

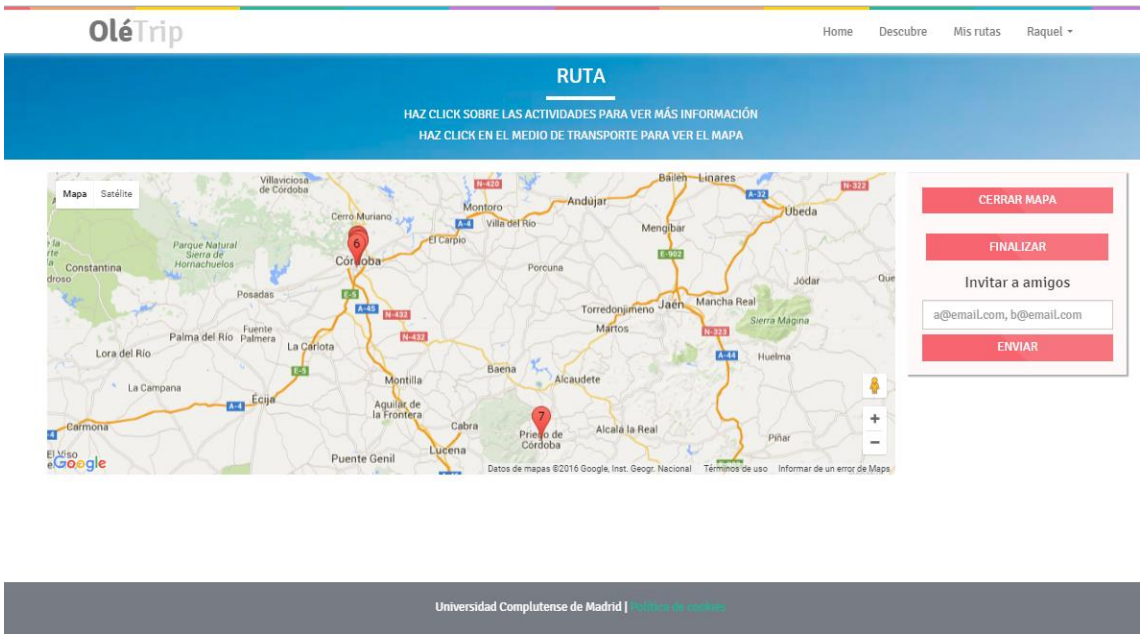
- Prioridad baja
  - Añadir actividades a la planificación

    Las actividades desde un principio podían ser arrastradas o añadidas con el botón de añadir, sin embargo, al no estar indicado en el "carrito" lo de añadir, algunos usuarios no intuían tal funcionalidad, por ese motivo, se añadió en el "carrito": o dar al botón de añadir.

- Envío de invitaciones

  Para agilizar el envío de invitaciones, se ha modificado el cuadro de texto para que sea posible introducir varios correos electrónicos, separados por coma, y enviar la invitación a todos con un sólo clic.

En definitiva, gracias a las evaluaciones realizadas, se han conseguido detectar y solventar varios problemas de usabilidad, desde el más mínimo al más prioritario, para así, en un futuro la aplicación pueda resultar más cómoda para el resto de usuarios. Aunque cabe destacar, que diez personas no representan a la mayoría, y puede haber más problemas aún no detectados.



6.15 EVALUACIÓN. CAMBIO 1

7. Conclusiones y visión futura

Implementar un sistema correctamente, y más uno tan extenso y complejo como éste, presenta muchos frentes que se pueden mejorar, como la optimización de recursos, de algoritmos, enfoque, etc. Ya que el mundo de las "guías turísticas online" está en auge, y llegar a conseguir un sistema de recomendación idóneo para todos es complejo, ya que cada viajero es un mundo.

Sin embargo, se ha llegado conseguir un sistema válido que ahorra mucho tiempo al viajero, ya que no tiene que calcular ni tiempos ni rutas, ya que eso es ofrecido por el sistema. El cual ha sufrido diversos cambios para poder ser lo que es finalmente, ya que, al ser una aplicación muy abierta, tiene muchas líneas de trabajo.

Se ha conseguido aportar una serie de actividades que se pueden realizar en las diferentes provincias de Andalucía, y poder incorporarlas en el viaje, así como poder valorarlas por los invitados del viaje, y si tienen valoraciones negativas, cambiarlas por otras más afines, hasta conseguir la satisfacción máxima del grupo.

Además, se ha realizado una evaluación con usuarios reales, los cuales nos han proporcionado retroalimentación del trabajo realizado. Tras la realización de dicha evaluación, se han podido añadir alguna funcionalidad (mapa general), así como corregir pequeñas deficiencias (detalles de lugares, ubicación de me gusta y no me gustas, etc.)

Por otra parte, cabe destacar que este trabajo deja muchas puertas abiertas, es decir, a raíz de esta aplicación pueden surgir otras muy similares, y con muchas mejoras. Quizás una buena alternativa al algoritmo de cálculo de ruta, sería basarla en poder siempre retornar al lugar de estancia durante el viaje, para ello sería ideal que la aplicación ofreciera recomendaciones de hoteles, casas rurales, apartamentos, etc., que estén cerca de los lugares a visitar, o que el usuario pueda indicar dónde se va a quedar.

Otra alternativa, es hacer que la aplicación crezca, es decir, que no sólo se puedan planear viajes por Andalucía, sino por otras ciudades de España, incluso llegando a ofrecer viajes por otros países europeos. Para ello, sólo bastaría con aumentar el catálogo de actividades existentes, añadiendo más lugares a la base de datos, y mejorando, quizás los datos existentes, y a su vez la estructura de la misma.

También, sería una buena idea, mejorar la obtención de preferencias por parte de los usuarios, para llegar a obtener una mejor ruta para él, que pueda llegar a ser más afín, quizás para ello sería mejor centrar la preferencia en cada día, o en cada provincia o ciudad a visitar, aumentando así las probabilidades de ser más satisfactorio.

Sin embargo, estos son algunos ejemplos de muchos, este tipo de aplicaciones tiene muchas vertientes, y pueden ser explotados más a fondo, consiguiendo sistemas más inteligentes.

8. Conclusion and future work

Implementing a system, large and complex like this one, shows many improvements that can be done, such as resource optimization, algorithms, focus, etc. Since the world of "online tourist guides" is booming, getting a suitable recommendation system for all participants is a complex work because every traveler is different.

However, we have achieved a proper system that could save a lot of time to travelers, because they do not have to calculate schedules or routes, since it is offered by the system. This system has suffered several changes in order to be what it is now, being a very open application that it has many lines of work.

We have managed to bring a number of activities that could be performed in different provinces of Andalusia, and that it is able to incorporate them in the journey, and rate them for the trip's participants. If those activities have negative ratings, the owner could change them by similar activities to get the maximum satisfaction from the group.

In addition, an evaluation has been made with real users, which have provided us feedback from the work done. Following the completion of this evaluation, the page has been able to show some new functionality (general map) and correct minor deficiencies (details of places, location, buttons, etc.).

On the other hand, we want to clarify that this work leaves many open doors, this means that following this application may appear other very similar, and with many improvements. Perhaps a better alternative to the algorithm for calculating route, would be to base it on returning to the place where the travelers stay during the trip, because of this, it would be ideal that the application offered recommendations for hotels, cottages, apartments, etc., which were close to the places to visit, or that the user could indicate where they are staying.

Another alternative is to expand the application, e.g. that not only plan trips in Andalusia could be planed, but also for other cities of Spain, or even be able to offer trips to other European countries. To do so, it would be enough to

increase the catalog of existing activities, adding more locations to the database, and improving, perhaps existing data.

It would also be a good idea to improve the collection of preferences by the users, achieving a better route, which may become more akin, another possibility could be to focus preferably on each day, each province or city to visit, increasing this way the chances of being more satisfactory.

Nonetheless, these are only a few examples, these kind of applications can be developed further, getting a more intelligent system.

9. Aportaciones

En este capítulo se expondrá el trabajo que ha realizado cada uno de los integrantes del proyecto. La asignación de trabajo fue en función de los intereses y gustos de cada uno, intentando siempre que fuese de forma equitativa.

9.1. Aportaciones de Raquel Álvarez

Como primera tarea, empecé creando los diseños de la interfaz, así como el de la base de datos, al igual que mis compañeros, y entre todos, y con los tutores, hemos decidido los diseños finales, por lo que han sufrido diversos cambios a lo largo del desarrollo del proyecto.

Tras ello, hemos decidido elegir las diversas tecnologías a emplear, y, empecé junto a mis compañeros a decidir los servicios que ofrecería el sistema (nombre, funcionalidad, etc.), una vez decididos, pasamos al diseño en java de los mismos. Yo, hasta que mis compañeros no crearon la estructura en java, no empecé. Cuando fue así, colaboré, en un principio, en establecer la manera de poder consumir y producir datos en JSON mediante los servicios, parte clave para el sistema, y a continuación, en la implementación de varios métodos, y servicios.

Al mismo tiempo, en base a los diseños de la interfaz, empecé a desarrollarla, partiendo de la creación de la estructura de directorios, archivos, etc., así como la búsqueda de assets, fuentes, imágenes, etc., qué posteriormente usé para el diseño final.

Cuando conseguimos que los servicios fuesen correctamente, empezamos a decidir entre todos, el algoritmo apropiado para el cálculo de rutas. Además, proporcioné diversas ideas en cuanto al funcionamiento de los servicios, y como mejorarlos a medida que estos iban siendo implementados.

A la vez que iba desarrollando, tanto el backend como el frontend, iba realizando pruebas de funcionamiento, y adaptabilidad (interfaz) en todos los dispositivos, así como todo cumpliera su cometido.

Cuando empezamos a configurar el servidor, mi compañero, encontró una serie de problemas, uno de ellos, el que atañía al MySQL fue solventado por mí, y esto fue clave para que el núcleo de la aplicación recabara los datos necesarios.

Las tareas realizadas pueden resumirse en:

- Diseño de los prototipos de la base de datos
- Diseño de mockups
- Maquetado de la interfaz
- Diseño de la interfaz
- Elección de frameworks y tecnologías para el frontend.
- Implementación de todas las funcionalidades de la interfaz
- Implementación del uso de JSON en el backend
- Implementación de varios servicios del backend
- Solventación de problemas MySql en el servidor de la universidad
- Afrontar los distintos problemas o errores que hubo a lo largo de todo el desarrollo entre las distintas capas (frontend, backend, BBDD)
- Propuestas de diseño delos algoritmos de planificación
- Testeo de todas las partes, en busca de errores y mejoras.

Tras elaborar todo el sistema, empezamos con la documentación, es decir, esta memoria, la cual la hemos elaborado conjuntamente, cada uno se dedicaba a una parte, sin embargo, yo revisaba cada parte, y en el caso de que hubiese algún fallo o faltase algo, era yo quien lo solventaba. Sin embargo, yo me centré más en la parte de interfaz, resumen, introducción, conclusiones, y en la investigación y estado del arte. Y entre todos, las traducciones de las partes requeridas. Además de ayudar en la creación de los formularios de la evaluación con usuarios, y aplicar las correcciones tras ella.

9.2. Aportaciones de Yanyan Cheng

He participado casi en todos los aspectos del proyecto. Durante la fase inicial tomé la iniciativa de la elaboración de los distintos bocetos de la interfaz como el diseño de la base de datos, el proyecto final se asemeja bastante a las ideas propuestas de mis prototipos. Para corregir y mejorar la base de datos, acudí con mi compañero Qiang Sun, al profesor de ampliación de base de datos, Manuel Montenegro.

Después de estos diseños iniciales hemos empezado a elegir las distintas tecnologías que íbamos a usar, una vez decido esto me he centrado principalmente en la parte de base de datos, en un principio, nuestro coordinador habló con una empresa externa y nos proporcionó una base de datos con información sobre sitios turísticos de la comunidad de Andalucía.

La base de datos nos ha dado varios problemas al cargarlo en MySQL ya que el formato no era de su tipo. Una vez resuelto vimos que le faltaba información que necesitamos a la hora de trabajar además de que existían muchos campos innecesarios, a partir de entonces, creé una base de datos nueva extrayendo de dicha base de datos solamente algunos campos necesarios como: coordenadas, nombre del lugar, id, dirección y pagina web correspondiente. He escogido los 100 sitios más populares y les he completado con las distintas informaciones que vamos a tratar en nuestro sistema.

Las distintas tareas que llevé a cabo en la creación de la base de datos fueron:

- Implementación de las distintas entidades de la capa JPA, llevando a cabo la creación de la base de datos acorde al diseño inicial definitivo.
- Extracción y búsqueda de los datos necesarios para completar los distintos campos de la tabla "sites", estos fueron, por ejemplo: nivel de interés, horas que duran las actividades, descripción, etc.

- Mantener y actualizar la base de datos con los nuevos requerimientos que fueron sugiriendo a lo largo de la evolución del proyecto.
- Elaboración de las distintas consultas en la capa DAO para las operaciones sobre estos datos.
- Busqué fotos identificativas de cada sitio para más ser mostrada más tarde, estas fotos no le hemos guardado en la misma base de datos, sino, en otra carpeta aparte que irá identificado cada foto por el id que tiene en la base de datos.

A parte de estos ayude también en la capa de negocio, ayude a diseñar los algoritmos de planificación de rutas, aportando conocimientos aprendidos en la asignatura de Aprendizaje Automático, contribuí también en la construcción de proyecto.

Comencé primero elaborando el borrador inicial de la memoria, consultando las distintas normativas para esta. Me enfoqué sobre todo en los puntos relacionados con la base de datos y otros puntos generales como: introducción, motivación, investigación, etc. Elaboré las preguntas de las entrevistas a los usuarios donde también realicé unas cuantas evaluaciones a mis familiares.

Dentro de la memoria también redacté ejemplo de casos de usos y descripción de cada una de ellas. Las traducciones de abstract y conclusión y visión futura lo hicimos entre todos juntos. Para llevar a cabo toda la memoria hemos trabajado colaborativamente, ya que tuvimos que editar cosas o mejorar cosas mutuamente.

9.3. Aportaciones de Qiang Sun

Durante la fase inicial participé en la elaboración de los distintos bocetos de la interfaz como el diseño de la base de datos. Los diseños difieren un poco de los bocetos iniciales ya que la hemos ido madurando y mejorando.

Tras ello empezamos a elegir las distintas tecnologías que íbamos a usar, una vez decido esto me he centrado principalmente en la parte de backend, a la vez que estudiaba el desarrollo de servicios web en java empecé elaborando una lista de servicios a crear, definiendo junto con las otras dos compañeras las URL de petición y el formato de respuestas.

Las distintas tareas que lleve a cabo en el desarrollo del backend fueron:

- Implementar la conexión entre servidor-cliente y la base de datos, estos fueron sumamente importante ya que el frontend dependía directamente de los servicios del que el *backend* es capaz de ofrecer y a la vez que la base de datos sea capaz de proveer y guardar todo lo necesario.
- Elegir los distintos patrones a seguir para que la aplicación fuera robusta y fácil de mantener, solucionando los problemas típicos y recurrentes. Estos fueron, por ejemplo: patrón de acceso DAO para base de datos, patrón Singleton para muchas instancias únicas como Log4j ó Hibernate, patrón arquitectónico de programación por capas, etc…
- Elección de los distintos framework necesarios, configurar el proyecto maven adecuadamente con los distintos artefactos y dependencias y configurar todo lo necesario para el correcto funcionamiento del proyecto web (web.xml y servlet).
- Implementación de los distintos algoritmos de la capa de negocio como el algoritmo núcleo de planificación, este algoritmo en un principio fue desarrollada con el algoritmo voraz, donde más tarde vimos la necesidad de mejorar aplicando el algoritmo de K-means para agrupamiento de sitios.

- Afrontar los distintos problemas o errores que hubo a lo largo de todo el desarrollo entre las distintas capas (frontend, backend, BBDD).

Aparte de estos he desarrollado varias funcionalidades como cargas de datos del fichero "oletrip.properties" donde se guarda las configuraciones básica del proyecto; funcionalidades de trazas de log en el fichero de texto "oletrip.log" para la verificación, depuración, control en todo el tiempo; solicitar correo Gmail para la aplicación y programar la clase "Email" que se encarga de enviar todas las invitaciones, entre muchas otras...

También ayude en diversos aspectos en las otras capas. Para el frontend he realizado las distintas pruebas en distintos dispositivos y pantallas para asegurar que la página sea usable en todas las plataformas y con distintas escalas. En cuanto BBDD participé en el arreglo de unos errores de inconsistencia de la capa JPA que causaba problemas al hacer algún borrado, tareas como mantenimientos y actualizaciones, elaboraciones de consultas HQL específicas.

Colaboré en configurar y solucionar los errores de los distintos entornos de trabajo para el desarrollo (Eclipse + servidor local) o servidor online proporcionado por los personales de laboratorio de nuestra facultad. Hemos empezando trabajando en local hasta poder desplegar el proyecto war al tomcat del servidor online. Como es el primer año que ofrecen nuestra facultad estos servidores, me ha dado bastantes problemas que resolver/enfrentar.

Para realizar la memoria hemos trabajado en conjunto, donde repartimos por puntos dependiendo de que había hecho cada miembro ya que dominaban mejor esa faceta, pero mucho de estos puntos se tuvo que rescribir o modificar del que finalmente acabamos casi editando o mejorando un poco de todo. Yo me centré más en redactar lo que era relacionado con las tecnologías y partes técnicas, alguno de los contenidos realizados por mi fueron, por ejemplos: Tabla de explicación de cada uno de los algoritmos, diagramas explicativos, imágenes ilustrativas de contenidos, diversas entrevistas a usuarios, etc.

APÉNDICES

A. Base de datos

a. *Tabla User*

Tabla de datos de perfil de usuarios.

Atributos	Tipo	Comentario
idUser	Int(11)	Clave primaria de la tabla
bornDate	date	Fecha de nacimiento
email	Varchar(255)	Correo electrónico
online	Bit(1)	Booleano para saber si está conectado
passwd	Varchar(255)	Contraseña del usuario
photo	longblob	Foto del perfil
shadowProfile	Bit(1)	Booleano para saber si está registrado o invitado
tokens	Varchar(255)	Elemento de seguridad

8 BASE DE DATOS. USER

b. *Tabla Trip*

Tabla de datos sobre un viaje.

Atributos	Tipo	Comentario
idTrip	Int(11)	Clave primaria de la tabla
definitive	Bit(1)	Booleano para saber si un viaje está cerrado, es decir que no admite más cambios
end	date	Fecha fin del viaje
rate	double	Rango de tiempo de visita, depende de si quiere ir rápido o relajado, valores: 1 ó 0.5
satisfaction	Int(11)	Sirve para evaluar el viaje, rango de valores de 1 a 5
start	date	Fecha inicio del viaje
Time_end	Int(11)	Hora fin de la última actividad del día en segundos
Time_start	Int(11)	Hora inicio de la primera actividad del día en segundos
transport	Varchar(255)	Transporte preferido por el usuario

Owner_idUser	Int(11)	Guarda la clave primaria del creador (para relación)

9 BASE DE DATOS. TRIP

c. *Tabla Opinion*

Tabla de datos de opiniones sobre actividades de un viaje.

Atributos	Tipo	Comentario
idOpi	Int(11)	Clave primaria de la tabla
voted	Bit(1)	Booleano para saber el voto
TripSite_idTripSites	Int(11)	Clave externa de la tabla Trip_sites
user_idUser	Int(11)	Clave externa de la tabla User

10 BASE DE DATOS. OPINION

d. *Tabla Sites*

Tabla de datos de los sitios.

Atributos	Tipo	Comentario
idSite	Int(11)	Clave primaria de la tabla
address	Varchar(11)	Dirección del sitio
city	Varchar(11)	Clave externa de la tabla Trip_sites
description	Varchar(11)	Clave externa de la tabla User
duration	double	Duración media de visita del sitio, en minutos
email	Varchar(255)	Correo del sitio turístico
name	Varchar(255)	Nombre del lugar
PhoneNum	Int(11)	Teléfono de contacto del sitio turístico
rating	Int(11)	Valoracion del sitio
subtype	Varchar(255)	Subcategoría del sitio
town	Varchar(255)	Población del sitio
type	Varchar(255)	Categoría del sitio
web	Varchar(255)	Página web del sitio
x	double	Coordenada de longitud
y	double	Coordenada de latitud

11 BASE DE DATOS. SITES

e. ***Tabla Trip_user***

Tabla de relación usuario y viaje.

Atributos	Tipo	Comentario
ListTrip_idTrip	Int(11)	Id del viaje
ListInvited_idUser	Int(11)	Id del usuario invitado

12 BASE DE DATOS. TRIP_USER

f. ***Tabla Trip_sites***

Tabla de relación de viaje y sitios.

Atributos	Tipo	Comentario
idTripSites	Int(11)	Clave primaria de la tabla
day	Int(11)	El dia del viaje
distanceNextSite	Int(11)	Distancia al siguiente sitio
durationNextSite	Varchar(255)	Tiempo empleado para llegar al siguiente sitio
durationSite	Varchar(255)	Tiempo empleado para estar en el sitio
time	Varchar(255)	Hora de comienzo de la visita
transport	Varchar(255)	Método de transporte
Site_idSite	Int(11)	Id del sitio
Trip_idTrip	Int(11)	Id del viaje

13 BASE DE DATOS. TRIP_SITES

g. ***Tabla Trip_preferences***

Tabla de relación de viaje y sitios.

Atributos	Tipo	Comentario
Trip_idTrip	Int(11)	Id del viaje
preference	Varchar(255)	Preferencia elegida por el usuario que se guardará en orden

14 BASE DE DATOS. TRIP_PREFERENCES

h. ***Tabla Trip_cities***

Tabla de relación de viaje y ciudades.

Atributos	Tipo	Comentario
Trip_idTrip	Int(11)	Id del viaje
cities	Varchar(255)	Ciudad a visitar

15 BASE DE DATOS. TRIP_CITIES

i. *Tabla Opinion_comment*

Tabla de relación de viaje y sitios.

Atributos	Tipo	Comentario
Opinion_idOpi	Int(11)	Clave primaria de la tabla opinión.
comment	Varchar(255)	Comentario del usuario
Comment_key	datatime	Clave primaria del opinión

16 BASE DE DATOS. OPINION_COMMENT

B. Servicios OléTrip

- ***User Services***
  - **Registro (POST)**

URL

<servidor>/TFG_OLETRIP/user/signup

Parámetros

email: correo electrónico del usuario.

name: nombre del usuario.

passwd: contraseña de acceso (mejor cifrada).

bornDate: fecha de nacimiento del usuario (en segundos).

Respuesta

MSG: mensaje del sistema (OK o ERROR).

Ejemplo

JSON con datos a enviar

```
{
      "email": "user@email.com",
      "name": "user",
      "passwd": "contraseña",
      "bornDate": 757036800
  }
```

Resultado

```
{
      "MSG": "OK"
}
```

  - **Login (POST)**

URL

<servidor>/TFG_OLETRIP/user/login

Parámetros

email: correo electrónico del usuario.

passwd: contraseña de acceso (mejor cifrada).

Respuesta

MSG: mensaje del sistema (OK o ERROR).

idUser: identificador de usuario.

tokens: tokens únicos de usuario.

username : nombre del usuario.

Ejemplo

JSON con datos a enviar

```
{
      "email": "user@email.com",
```

```
      "passwd": "contraseña"
}
Resultado
{
      "MSG": "OK",
      "idUser": 1,
      "tokens": "khaf561772jjd833ijd9",
      "username": "user"
}
```

- **Datos de usuario (GET)**

URL

```
<servidor>/TFG_OLETRIP/user/{id_user}
```

Parámetros

```
Ninguno
```

Respuesta

```
name: nombre del usuario.
bornDate: fecha de nacimiento del usuario.
email: email del usuario.
opinion: lista de identificadores de opinión del usuario.
createdTrip: lista de identificadores de viajes creados por el usuario.
listTrip: lista de identificadores de viajes a los que ha sido invitado el usuario.
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/user/1
Resultado
{
  "name": "user",
  "bornDate": "1993-12-28",
  "email": "user@email.com",
  "opinion": "[1, 2]".
  "createdTrip": "[1, 2, 6]",
  "listTrip": "[5]",
}
```

- **Modificación de datos de usuario (POST)**

URL

```
<servidor>/TFG_OLETRIP/user/{id_user}/update
```

Parámetros

```
Obligatorios
      tokens: tokens únicos de usuario (seguridad).
Opcionales
      name: nombre de usuario nuevo.
      bornDate: fecha de nacimiento a modificar.
      email: email de usuario nuevo.
      passwd: nueva contraseña.
```

Respuesta

MSG: mensaje del sistema (OK o ERROR).
userID: si es OK devuelve el id del usuario.

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/user/1/update
JSON con datos a enviar
{
      "tokens":"khaf561772jjd833ijd9",
      "name": "Raquel"
}
Resultado
{
      "MSG": "OK",
      "userID": 1
}
```

- **Cierre de sesión (POST)**

URL

<servidor>/TFG_OLETRIP/user/{id_user}/logout

Parámetros

tokens: tokens únicos de usuario (seguridad).

Respuesta

true o false

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/user/1/logout
JSON con datos a enviar
      khaf561772jjd833ijd9
Resultado
      true
```

- ***Trip Services***
  - **Creación de un viaje (POST)**

URL

<servidor>/TFG_OLETRIP/trip/create

Parámetros

end: fecha de fin del viaje.

start: fecha de inicio del viaje.

tokens: tokens únicos de usuario (seguridad).
cities: provincias a visitar.

preferences: categorias de viaje en orden de preferencia.

rate: modo de viaje (1 o 0.5).

transport: medio de transporte (“driving” o “transit”).

time_start: hora de inicio de actividades en segundos.

time_end: hora de fin de actividades en segundos.

Respuesta

MSG: mensaje del sistema (OK o ERROR).

idTrip: identificador del viaje

Ejemplo

Url de acceso

<servidor>/TFG_OLETRIP/trip/create

JSON con datos a enviar

```
{
      "end":"2016-07-03",
      "start":"2016-06-30",
      "tokens":"khaf561772jjd833ijd9",
      "cities":["almeria","cadiz"],
      "preferences":["Arquitectura","Monumento","Mus
      eo","Cultura","Ocio","Playa","Naturaleza","Bod
      ega","Deporte","Parque"],
      "rate":"0.5",
      "transport":"driving",
      "time_start":32400,
      "time_end":75600
      }
```

Resultado

```
{
  "MSG": "OK",
  "idTrip": 8
}
```

- **Creación de la ruta (POST)**

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/rute

<servidor>/TFG_OLETRIP/trip/{id_trip}/rute_1

Parámetros

tokens: tokens únicos de usuario (seguridad).

lugares: lista de identificadores de lugares a visitar.

start: identificador del primer lugar a visitar (sólo para rute)

Respuesta

```
{ "day_n":[
            {
                  "activity_n":  {
                  }, ...
            },
            {
                  "lunch_duration": 3600000,
                  "lunch_time": 50400000
            }
      ],
      ...
}
```

Ejemplo

Url de acceso

<servidor>/TFG_OLETRIP/trip/8/rute_1

JSON con datos a enviar


```
{
      "tokens": "khaf561772jjd833ijd9",
      "lugares":
      [5418,5220,5206,5085,6764,6744,8764,8780,5419,
      5203,7931,5423,7787]
}
```


Resultado


```
{
  "day_1": [
    {
      "activity_3": {
        "duration": 1800000,
        "distance": 34486,
        "city": "Almeria",
        "idSite": 5085,
        "name": "La Alcazaba",
        "x": 36.8405997102619,
        "y": -2.47079253930024,
        "time": 43200000,
        "timeActivity": 2700000,
        "tranport": "driving"
      },
      "activity_2": {
        "duration": 900000,
        "distance": 2356,
        "city": "Almeria",
        "idSite": 6764,
        "name": "Museo Arqueológico de Almeria",
        "x": 36.8384759961096,
        "y": -2.45533950854965,
        "time": 38700000,
        "timeActivity": 1800000,
        "tranport": "driving"
      },
      "activity_1": {
        "duration": 900000,
        "distance": 1648,
        "city": "Almeria",
        "idSite": 5418,
        "name": "Plaza de Toros ",
        "x": 36.8468162796282,
        "y": -2.4616274963183,
        "time": 32400000,
        "timeActivity": 2700000,
        "tranport": "driving"
      },
      "activity_4": {
        "duration": 2700000,
        "distance": 44992,
```

```
            "city": "Almeria",
            "idSite": 5419,
            "name": "Mini Hollywood Poblado del Oeste",
            "x": 37.0209906187266,
            "y": -2.43405411945885,
            "time": 54000000,
            "timeActivity": 7200000,
            "tranport": "driving"
          }
        },
        {
          "lunch_duration": 3600000,
          "lunch_time": 50400000
        }
      ],

      . . . así con todos los días
    }
```

- **Guardar la ruta (POST)**

URL

```
<servidor>/TFG_OLETRIP/trip/{id_trip}/save
```

Parámetros

```
tokens: tokens únicos de usuario (seguridad).
Lista de dias, cada uno con la siguiente estructura:
{
      "day": 1,
      "time":"32400000",
      "durationNextSite": 900000,
      "distanceNextSite": 1648,
      "siteId":"5418",
      "durationActivity":"2700000",
      "mode":"driving"
}
```

Respuesta

```
MSG: Mensaje del sistema (OK o ERROR)
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/save
JSON con datos a enviar
{
      "tokens":"khaf561772jjd833ijd9",
      "guardar”: [
      {
            "day":1,
            "time":"32400000",
            "durationNextSite":"900000",
            "distanceNextSite":"1648",
            "siteId":"5418",
```

```
            "durationActivity":"2700000",
            "mode":"driving"
            },
            {
            "day":1,
            "time":"38700000",
            "durationNextSite":"900000",
            "distanceNextSite":"1108",
            "siteId":"6764",
            "durationActivity":"1800000",
            "mode":"driving"
            }, ... así con todas las actividades en
            orden
      ]
}
Resultado
      "MSG": "OK"
```


- **Recuperación de ruta (GET)**

URL

`<servidor>/TFG_OLETRIP/trip/{id_trip}`

Parámetros

Ninguno

Respuesta

owner: creador del viaje.

preferences: preferencias del viaje.

cities: provincias del viaje.

start: fecha de inicio.

end: fecha de fin.

definitive: viaje definitivo o no.

trip_sites: JSON con las actividades de la ruta, ordenadas por días y horas.

transport: medio de transporte.

score: puntuación del viaje.

idTrip: identificador del viaje.

rate: modo de viaje.

dias: número de días.

invitedUser: lista de invitados

Ejemplo

Url de acceso

`<servidor>/TFG_OLETRIP/trip/8`

Resultado


```
{
  "owner": 1,
  "preferences": [
    "Arquitectura",
```

```
    "Monumento",
    "Museo",
    "Cultura",
    "Ocio",
    "Playa",
    "Naturaleza",
    "Bodega",
    "Deporte",
    "Parque"
  ],
  "cities": [
    "almeria",
    "cadiz"
  ],
  "start": "2016-06-30",
  "end": "2016-07-03",
  "definitive": false,
  "trip_sites": {
      (RUTA)
  },
  "transport": "driving",
  "score": 0,
  "idTrip": 8,
  "rate": 0.5,
  "dias": 4,
  "invitedUser": "[]"
}
```

- **Ruta definitiva (POST)**

  URL

  `<servidor>/TFG_OLETRIP/trip/{id_trip}/definitive`

  Parámetros

  `tokens: tokens únicos de usuario (seguridad).`

  Respuesta

  `True o false`

  Ejemplo

  `Url de acceso`
  `<servidor>/TFG_OLETRIP/trip/8/definitve`
  `JSON con datos a enviar`
  `khaf561772jjd833ijd9`
  `Resultado`
  `true`

- **Creador**
  - **GET**

    URL

    `<servidor>/TFG_OLETRIP/trip/{id_trip}/owner`

    Parámetros

    `ninguno`

Respuesta

owner: email del creador

id: id del creador

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/owner
Resultado
{
  "owner": user@email.com
  "id": 1
}
```

- **POST**

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/owner

Parámetros

tokens: tokens únicos de usuario (seguridad).

Respuesta

True o false

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/owner
JSON con datos a enviar
      khaf561772jjd833ijd9
Resultado
      true
```

- **Invitar (POST)**
  - **Crear invitación**

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/invite

Parámetros

tokens: tokens únicos de usuario (seguridad) del creador del viaje.
name: nombre del usuario a invitar.
email: correo electrónico del usuario a invitar.

Respuesta

MSG: mensaje del sistema (OK o ERROR)

id: si es Ok, devuelve el id del usuario invitado.

tokens: si es OK, devuelve los tokens de usuario invitado.

Ejemplo

Url de acceso

<servidor>/TFG_OLETRIP/trip/8/invite

JSON con datos a enviar

```
{
    "tokens":          "khaf561772jjd833ijd9",
    "name":"user2",
    "email":"user2@email.com",
}
```

Resultado

```
{
  "id": 2,
  "tokens": "dafs772nc8ayaf34vdsgadg"
  "MSG": "OK"
}
```

- **Enviar invitación**

URL

`<servidor>/TFG_OLETRIP/trip/{id_trip}/invited/{id_user}/send`

Parámetros

`tokens: tokens únicos de usuario (seguridad) del creador del viaje.`
`name: nombre del remitente.`
`url: url de acceso a ver el viaje.`

Respuesta

`MSG: mensaje del sistema (OK o ERROR)`

Ejemplo

`Url de acceso`
`<servidor>/TFG_OLETRIP/trip/8/invited/2/send`
`JSON con datos a enviar`

```
{
    "tokens":         "khaf561772jjd833ijd9",
    "name":"OleTrip",
    "url":"
    http://oletrip.esy.es/ver.html?id=8&tokens=2- dafs772nc8ayaf34vdsgadg "
}
```

```
Resultado
{
  "MSG": "OK"
}
```

- **Invitados**
  - GET

URL

`<servidor>/TFG_OLETRIP/trip/{id_trip}/invited`

Parámetros

`ninguno`

Respuesta

```
id: email
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/invited
Resultado
{
      "2": "ralvhdez@gmail.com"
}
```

- POST

URL

```
<servidor>/TFG_OLETRIP/trip/{id_trip}/invited
```

Parámetros

```
tokens: tokens únicos de usuario (seguridad) del
invitado a verificar si es o no.
```

Respuesta

```
True o false
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/invited
JSON con datos a enviar
      031a39d74c6443978b8eeace64d9863d
Resultado
      True
```

- Valoración del viaje
  - GET

URL

```
<servidor>/TFG_OLETRIP/trip/{id_trip}/vote
```

Parámetros

```
ninguno
```

Respuesta

```
int entre 0-5
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/vote
Resultado
      0
```

  - POST

URL

```
<servidor>/TFG_OLETRIP/trip/{id_trip}/vote
```

Parámetros

```
tokens: tokens únicos de usuario (seguridad) del
creador del viaje.
```

voto: int entre 0-5

Respuesta

True o false

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/vote
JSON con datos a enviar
{
      "tokens":"khaf561772jjd833ijd9",
      "voto":"4"
}
Resultado
      True
```

- **Eliminar un viaje (POST)**

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/delete

Parámetros

tokens: tokens únicos de usuario (seguridad) del creador del viaje.

Respuesta

True o false

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/delete
JSON con datos a enviar
      khaf561772jjd833ijd9
Resultado
      True
```

- **Eliminar invitación (POST)**

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/user/{id_user}/noinvite

Parámetros

tokens: tokens únicos de usuario (seguridad) del creador del viaje o del propio invitado.

Respuesta

True o false

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/8/user/2/noinvite
JSON con datos a enviar
      khaf561772jjd833ijd9
Resultado
      True
```

- **Recomendación de viajes (POST)**

  URL

  <servidor>/TFG_OLETRIP/trip/recommend

  Parámetros

  Lista de provincias

  Respuesta

  result_n : JSON similiar al de recuperación de un viaje.

  Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/recommend
JSON con datos a enviar
      ["sevilla"]
Resultado
{
  "result_1": {
    "cities": [
      "sevilla"
    ],
    "idTrip": 1,
    "rate": 1,
    "dias": 3,
    "trip_sites": {
            Itineriario del viaje
     },
    "transport": "transit"
  }
}
```

- ***Activity Services***
  - **Resumen de actividad (GET)**

    URL

    <servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}

    Parámetros

    ninguno

    Respuesta

    idTripSite: identificador de la actividad.
    idTrip: identificador del viaje.
    ciudad: provincias a visitar.
    idSite: identificador del lugar al que hace referencia.
    durationNext: tiempo empleado para ir a la siguiente actividad.
    name: nombre de la actividad.
    description: descripción de la actividad.
    duracion: duración de la actividad.
    time: hora de la actividad.

day: dia en el que se realiza la actividad.
distanceNext: distancia a la siguiente actividad.
url: url de referencia.

Ejemplo

Url de acceso
<servidor>/TFG_OLETRIP/trip/8/activity/79
Resultado


```
{
	"idTripSite": 79,
	idTrip": 8,
	"ciudad": "Almeria",
	"idSite": 7787,
	"durationNext": 3600000,
	"name": "Parque Natural del Cabo de Gata - Níjar",
	"description": "Es uno de los espacios naturales españoles, afectado por mayor número de figuras de protección, tanto de carácter natural como cultural. Estas inscripciones están fundamentadas en su riqueza geológica, ecológica, histórica, antropológica y paisajística",
	"duracion": 3600000,
	"time": 32400000,
	"day": 2,
	"distanceNext": 79871,
	"url": "www.cabogataalmeria.com"
}
```


- **Votación de actividad (POST)**

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}/vote

Parámetros

tokens: tokens únicos de usuario (seguridad) del creador o de un invitado.
voto: 1 ó -1

Respuesta

MSG: Mensaje del sistema (OK o ERROR)

idOpi: si es Ok, devuelve el identificador del voto.

Ejemplo

Url de acceso
<servidor>/TFG_OLETRIP/trip/8/activity/79/vote
JSON con datos a enviar


```
{
	"tokens":"khaf561772jjd833ijd9",
	"voto":"1"
```

```
}
Resultado
{
      "MSG":"OK",
      "idOpi":"3"
}
```

- **Comentarios**
  - GET

    URL

    <servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}/comments

    Parámetros

    ninguno

    Respuesta

    Numero de comentario: JSOn con estructura similar a la siguiente:

```
"1": {
    "comments": {
      "fecha": "comentario"
    },
    "userID": id,
    "email": email
}
```

    Ejemplo

    Url de acceso

    <servidor>/TFG_OLETRIP/trip/1/activity/15/comments

```
Resultado
{
  "1": {
    "comments": {
      "2016-05-09 18:10:29.0": "que tal",
      "2016-05-09 18:10:26.0": "mola"
    },
    "userID": 1,
    "email": "ra@gmail.com"
  }
}
```

  - POST

    URL

    <servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}/comments

    Parámetros

    tokens: tokens únicos de usuario(seguridad) del creador o de un invitado.

comentario: texto a expresar.

Respuesta

MSG: Mensaje del sistema (ERROR).

id: identificador del comentario.

email: email del usuario que comentó.

comment: comentario emitido.

Ejemplo

Url de acceso

<servidor>/TFG_OLETRIP/trip/1/activity/15/comments

JSON con datos a enviar

```
{
	"tokens": "adghakdgjghdklj",
	"comentario": "MOLA"
}
```

Resultado

```
{
	"id": "4",
	"email":"e@e.com",
	"comment":"MOLA"
}
```

- **Voto de un usuario (POST)**

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}/user/{id_user}/vote

Parámetros

Tokens: tokens únicos de usuario del usuario que se quiere verificar su votación.

Respuesta

MSG: mensaje del sistema (OK o ERROR).

value: 1 o -1 o 0 (me gusta, no me gusta, sin voto).

Ejemplo

Url de acceso

<servidor>/TFG_OLETRIP/trip/1/activity/15/user/vote

JSON con datos a enviar

Dgjslghaogiasogij1324

Resultado

```
{
	"MSG": "OK",
	"value": "1"
}
```

- **Me gusta y no me gusta (GET)**

URL

Me gusta:
<servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}/likes
No me gusta:
<servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}/dislikes

Parámetros

ninguno

Respuesta

Int con el número total de me gusta o no me gusta, dependiendo de la llamada.

Ejemplo

Url de acceso
<servidor>/TFG_OLETRIP/trip/1/activity/15/like
Resultado
8

- **Alternativa a una actividad y modificación**
  - Alternativa (GET)

URL

<servidor>/TFG_OLETRIP/trip/{id_trip}/activity/{id_act}/alternative

Parámetros

ninguno

Respuesta

Alternative_n: JSON similar a la de información de un lugar.

Ejemplo

Url de acceso
<servidor>/TFG_OLETRIP/trip/1/activity/15/alternative
Resultado


```
{
    "alternative_1": {
        "city": "Sevilla",
        "name": "Alcázar Real De Carmona",
        "x": 37.4728422031117,
        "siteId": 4969,
        "description": "El Real Alcázar de Sevilla es un conjunto de palacios rodeados por una muralla, situados en la ciudad de Sevilla. Su construcción se inició en la Alta Edad Media",
        "activityDuration": 5400000,
        "y": -5.63294534006494,
```

```
            "url":
            "http://www.turismo.carmona.org/tur
            istavirtual.htm"
      },
      "alternative_2": {
            "city": "Sevilla",
            "name": "Cruceros Turísticos",
            "x": 37.3693238174967,
            "siteId": 5339,
            "description": "Navegación por el
            Guadalquivir visitando la Sevilla
            clásica y la Sevilla moderna,
            ampliamente comentada en español,
            inglés, francés, portugués, alemán
            e italiano, el paseo por el río
            completará tu imagen de Sevilla.",
            "activityDuration": 14400000,
            "y": -5.99143912416179,
            "url":
            "http://www.visitasevilla.es/"
       }
}
```


- Modificación (POST)

URL

```
<servidor>/TFG_OLETRIP/trip/{id_trip}/activity
/{id_act}/update
```

Parámetros

```
tokens: tokens únicos de usuario (seguridad) del
creador del viaje.
siteId: identificador del lugar de intercambio
```

Respuesta

```
MSG: Mensaje del sistema (OK o ERROR)
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/trip/1/activity/15
      /update
JSON con datos a enviar
{
      "tokens":"tokens que sean",
      "siteId": 5339
}

Resultado
{
      "MSG":"OK"
}
```

- ***Site Services***
  - **Información de un lugar (GET)**

URL

<servidor>/TFG_OLETRIP/site/{site_id}

Parámetros

ninguno

Respuesta

address: dirección.

city: provincia,

description: descripción.

duration: duración en minutos,

email: email.

idSite: identificador.

name: Nombre de la actividad.

phoneNum: teléfono de contacto

rating: calificación de la actividad.

subtype: subcategoría a la que pertenece.

town: ciudad a la que pertenece.

type: categoría.

web: sitio web.

x: latitud

y: longitud

Ejemplo

Url de acceso

<servidor>/TFG_OLETRIP/site/4831

Resultado:


```
{
      "address": "Calle La Ermita, 1 - 21750, Almonte
      (Huelva)",
      "city": "Huelva",
      "description": "La ermita del Rocío es una
      ermita situada en la aldea de El Rocío, Almonte.
      En ella se celebra anualmente la famosa Romería
      de El Rocío, y que en la actualidad congrega a
      más de un millón de personas.",
      "duration": 90,
      "email": "",
      "idSite": 4831,
      "name": "Ermita Del Rocio",
      "phoneNum": 959442425,
      "rating": 4,
      "subtype": "Arquitectura Religiosa",
       "town": "Almonte",
      "type": "Arquitectura",
      "web":    "http://www.andalucia.org/es/turismo-
      cultural/visit",
```

```
        "x": -6.48460688059128,
        "y": 37.1307354375628
    }
```

- **Lugares pertenecientes a una provincia (GET)**

URL

```
<servidor>/TFG_OLETRIP/site/city/{provincia}
```

Parámetros

```
ninguno
```

Respuesta

```
Identificador y nombre del lugar
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/site/city/sevilla
Resultado
{
  "4865": "Plaza de España",
  "4875": "Torre Del Oro",
  "4885": "Catedral De Sevilla",
  "4904": "Iglesia Del Salvador",
  "4906": "Palacio De Lebrija",
  "4915": "Columas De La Alameda De Hércules",
  "4969": "Alcázar Real De Carmona",
  "5339": "Cruceros Turísticos",
  "6662": "Puerto Deportivo Gelves",
  "6695": "Museo Militar Regional",
  "6702": "Museo de Bellas Artes",
  "7624": "Teatro Lope de Vega",
  "7738": "Parque de María Luisa",
  "7748": "Parque El Alamillo",
  "7902": "Real Club Pineda de Sevilla",
  "7926":  "Reserva  Natural   El  Castillo  de  las
Guardas",
  "7927": "Aquópolis-Guadalpark",
  "7928": "Isla Mágica",
  "8594": "Archivo de Indias de Sevilla",
  "8811": "Centro Histórico de Sevilla"
}
```

- **Lugares pertenecientes a una provincia y categoría (GET)**

URL

```
<servidor>/TFG_OLETRIP/site/city/{provincia}/categor
y/{categoria}
```

Parámetros

```
ninguno
```

Respuesta

```
Identificador y nombre del lugar
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/site/city/Sevilla/categ
ory/museo
Resultado
{
  "4906": "Palacio De Lebrija",
  "6695": "Museo Militar Regional",
  "6702": "Museo de Bellas Artes"
}
```

- **Lugares pertenecientes a una categoría (GET)**

URL

```
<servidor>/TFG_OLETRIP/site/category/{categoria}
```

Parámetros

```
ninguno
```

Respuesta

```
Identificador y nombre del lugar
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/site/category/museo
Resultado
{
  "4906": "Palacio De Lebrija",
  "5401": "Parque de las Ciencias",
  "6691": "Parque Minero",
  "6695": "Museo Militar Regional",
  "6702": "Museo de Bellas Artes",
  "6712": "Museo de Bellas Artes",
  "6736": "Museo de Artes y Costumbres Populares",
  "6744": "Museo de las Cortes de Cadiz",
  "6749": "Museo de Artes y Costumbres Populares",
  "6752": "Museo Picasso",
  "6764": "Museo Arqueológico de Almeria"
}
```

- **Categorías de una provincia (GET)**

URL

```
<servidor>/TFG_OLETRIP/site/{provincia}/categories
```

Parámetros

```
ninguno
```

Respuesta

```
Lista de categorías pertenecientes a tal provincia
```

Ejemplo

```
Url de acceso
      <servidor>/TFG_OLETRIP/site/Sevilla/categories
Resultado
[
```

```
        "Arquitectura", "Museo", "Monumento", "Playa",
        "Cultura", "Parque", "Deporte", "Ocio"
    ]
```

- ***Other Services***
  - **Categorías (GET)**

    URL

    ```
    <servidor>/TFG_OLETRIP/service/category
    ```

    Parámetros

    ```
    ninguno
    ```

    Respuesta

    ```
    Lista de categorías totales
    ```

    Ejemplo

    ```
    Url de acceso
        <servidor>/TFG_OLETRIP/service/category
    Resultado
    [
        "Arquitectura", "Monumento", "Museo",
        "Cultura", "Ocio", "Playa", "Naturaleza",
        "Bodega", "Deporte", "Parque"
    ]
    ```

  - **Provincias (GET)**

    URL

    ```
    <servidor>/TFG_OLETRIP/service/cities
    ```

    Parámetros

    ```
    ninguno
    ```

    Respuesta

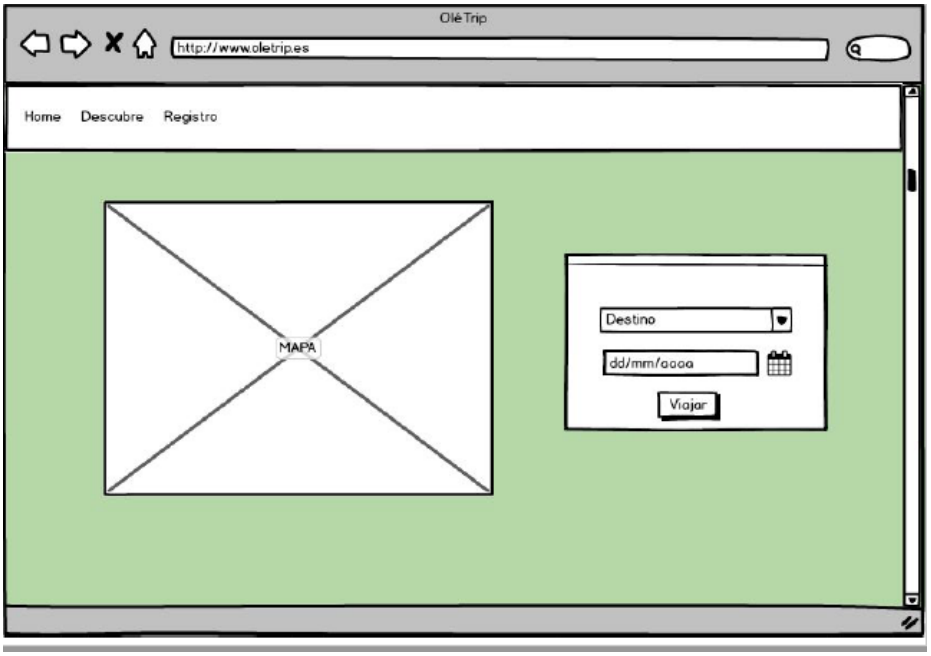
    ```
    Lista de todas las provincias
    ```

    Ejemplo

    ```
    Url de acceso
        <servidor>/TFG_OLETRIP/service/cities
    Resultado
    [
        "Huelva", "Malaga", "Sevilla", "Cordoba",
        "Granada", "Almeria", "Jaen", "Cadiz"
    ]
    ```

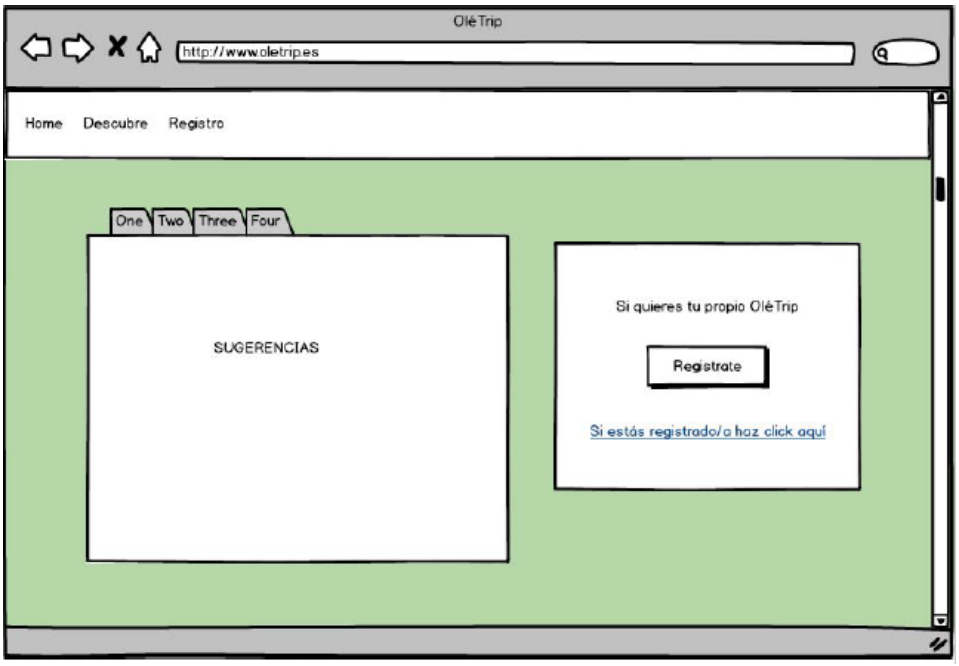
C. Mockups

a) Página de inicio



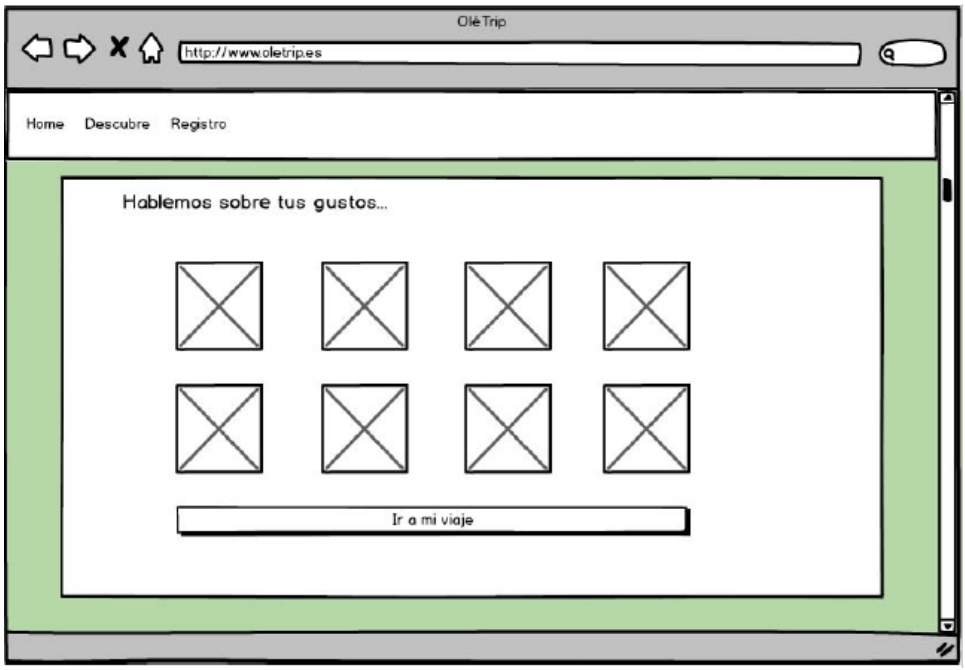
9.1 MOCKUPS. PÁGINA DE INICIO

b) *Página de sugerencias*



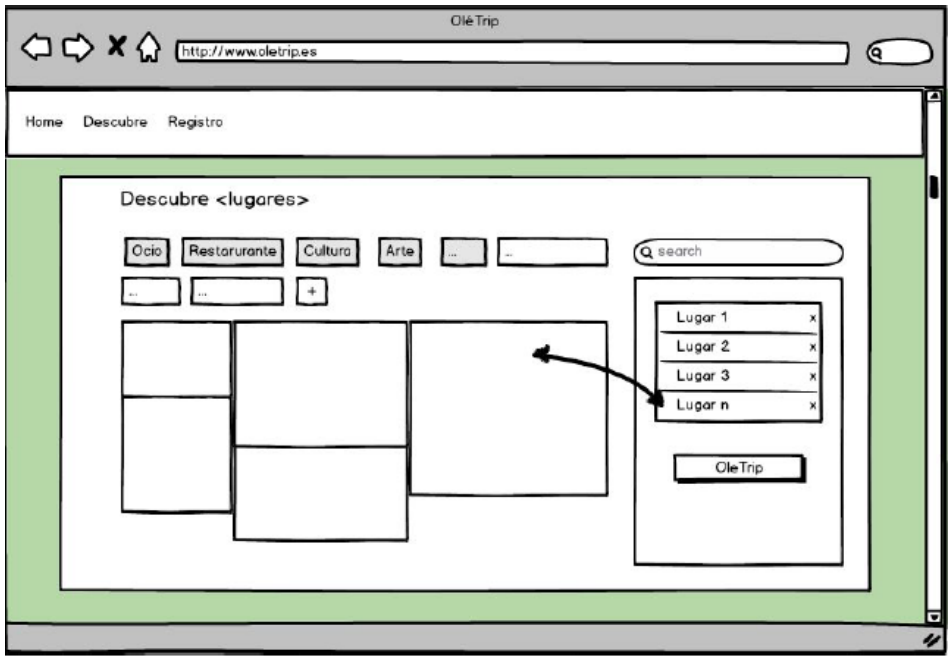
9.2 MOCKUPS. PÁGINA DE SUGERENCIAS

c) Página de preferencias



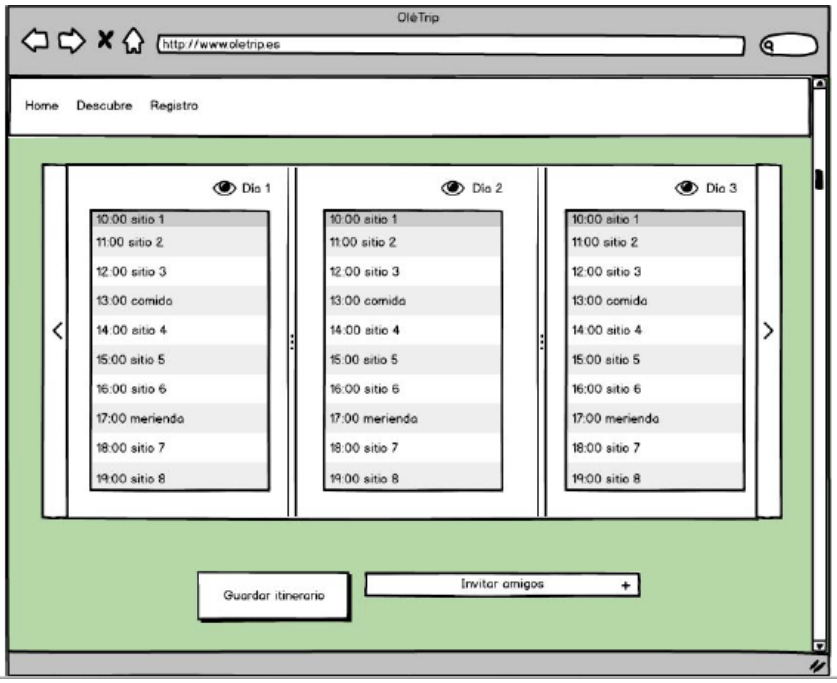
9.3 MOCKUPS. PÁGINA DE PREFERENCIAS

d) *Página de selección de lugares*



9.4 MOCKUPS. SELECCIÓN DE LUGARES

e) Página de previsualización de viaje (creador)



9.5 MOCKUPS. PREVISUALIZACIÓN DE VIAJE (CREADOR)

f) Página de previsualización de viaje (invitados)

9.6 MOCKUPS. PREVISUALIZACIÓN DE VIAJE (INVITADOS)

D. Configuración del entorno

a) *Configuración de dependencias de Maven*

GROUPID	ARTIFACTID	VERSIÓN
Com.owlike	Genson	0.99
Org.json	Json	20140107
Log4j	Log4j	1.2.17
Mysql	Mysql-connector-java	5.1.34
Org.hibernate	Hibernate-core	4.3.8.Final
Org.hibernate	Hibernate-c3p0	4.3.8.Final
Org.apache.commons	Commons-lang3	3.3.2
Org.glassfich.jersey.core	Jersey-client	2.22.2
Org.glassfish.jersey.containers	Jersey-container-servlet-core	2.5
Org-glassfish.jersey.bundles	Jaxrs-ri	2.5
Javax.servlet	Servlet-api	2.5
Asm	Asm	3.3.1
Org.apache.commons	Commons-io	1.3.2
Com.thetransactioncompany	Java-property-utils	1.9.1
Com.thetransactioncompany	Cors-filter	1.9.1
Javax.mal	Mail	1.4.7
Javax.mal	Javax.mail-api	1.5.2

17. CONFIGURACIÓN. MAVEN

b) *Configuración de Jersey (web.xml)*

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://java.sun.com/xml/ns/javaee"
      xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
      id="WebApp_ID" version="3.0">
      <display-name>OLETRIP</display-name>
      <welcome-file-list>
            <welcome-file>index.html</welcome-file>
      </welcome-file-list>
      <servlet>
            <servlet-name>TFG_OLETRIP</servlet-name>
            <servlet-class>
               org.glassfish.jersey.servlet.ServletContainer
      </servlet-class>
```

```
            <init-param>
                  <param-name>
                jersey.config.server.provider.packages
            </param-name>
                  <param-value>com.oletrip.webservice</param-value>
            </init-param>
            <load-on-startup>1</load-on-startup>
      </servlet>
      <servlet-mapping>
            <servlet-name>TFG_OLETRIP</servlet-name>
            <url-pattern>/*</url-pattern>
      </servlet-mapping>
      <listener>
            <listener-class>

com.oletrip.webservice.init.HibernateSessionFactoryListener
      </listener-class>
      </listener>
      <filter>
            <filter-name>
                CORS
      </filter-name>
            <filter-class>
           com.thetransactioncompany.cors.CORSFilter
      </filter-class>
      </filter>
      <filter-mapping>
            <filter-name>CORS</filter-name>
            <url-pattern>/*</url-pattern>
      </filter-mapping>
</web-app>
```

18 CONFIGURACIÓN. JERSEY

c) *Configuración de Log4j (log4j.properties)*

NOMBRE DE LA PROPIEDAD	VALOR
log4j.rootLogger	INFO, file
log4j.appender.file	org.apache.log4j.RollingFileAppender
log4j.logger.org.hibernate.SQL	OFF
log4j.logger.org.hibernate	WARN
log4j.logger.com.mchange.v2	WARN
log4j.appender.file.File	/tomcat/logs/oletrip.log
log4j.appender.file.MaxFileSize	10MB
log4j.appender.file.MaxBackupIndex	10

log4j.appender.file.layout	org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern	%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n

19 CONFIGURACIÓN. LOG4J

d) ***Configuración de properties de OléTrip (config.properties)***

NOMBRE DE LA PROPIEDAD	VALOR
url	[url del Google Map Matrix Api]
key	[clave para realizar peticiones]
mealId	[id asignado a la hora de comida]
email	[correo maestro]
epasswd	[contraseña del correo]

20 CONFIGURACIÓN. PROPERTIES

e) ***Configuración de MySQL server (my.conf y usuario de acceso):***

PROPIEDAD	VALOR
Usuario	Root
Contraseña	**********
Puerto cliente	3306
Socket cliente	/var/run/mysqld/mysqld.sock
Dirección base	/usr
Dirección datos	/var/lib/mysql

21 CONFIGURACIÓN. MYSQL SERVER

f) ***Configuración de Tomcat (archivo server.xml):***

PROPIEDAD	VALOR
Hostname	localhost
appBase	webapps
unpackWARs	true
Connector port	80
Connector redirectPort	8443

AJP 1.3 connector port	8009
AJP 1.3 redirectPort	8443

Bibliografía

1. Balabanović, Marko; Shoham, Yoav; "Fab: content-based, collaborative recommendation", Communications of the ACM,40,3,66-72, 1997, ACM
2. Jannach, D. (2008). Finding preferred query relaxations in content-based recommenders. In *Intelligent Techniques and Tools for Novel System Architectures* (pp. 81-97). Springer Berlin Heidelberg
3. Nieto, S. M. G. (2007). Filtrado colaborativo y sistemas de recomendación. Inteligencia de Redes de Comunicaciones, Universidad Carlos III de Madrid, Madrid.
4. Kolodner, J. (2014). Case-based reasoning. Morgan Kaufmann.
5. Burke, R. (1999, July). Integrating knowledge-based and collaborative-filtering recommender systems. In Proceedings of the Workshop on AI and Electronic Commerce (pp. 69-72).
6. Pazzani, M. J. (1999). A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review, 13(5-6), 393-408.
7. Burke, R. (2002). Hybrid recommender systems: Survey and experiments.User modeling and user-adapted interaction, 12(4), 331-370.
8. Masthoff, J. (2011). Group recommender systems: Combining individual models. In Recommender systems handbook (pp. 677-702). Springer US.
9. O'connor, M., Cosley, D., Konstan, J. A., & Riedl, J. (2001). PolyLens: a recommender system for groups of users. In ECSCW 2001 (pp. 199-218). Springer Netherlands.
10. Leiva, J. L., Guevara, A., Rossi, C., & Aguayo, A. (2014). Realidad aumentada y sistemas de recomendación grupales: Una nueva perspectiva en sistemas de destinos turísticos. Estudios y perspectivas en turismo, 23(1), 40-59.
11. Delgado, Cristina (2015). España tiene el sector turístico más competitivo del mundo. En **El País.**
12. Nayyar, Sarita (2016). Digital Media and Society: Implications inr a Hyperconnected Era. **World Economic Forum.**
13. Sistemas de recomendación en wikipedia

14. Java Platform, Standard Edition (2016). Documentación de especificación Java.
15. Oracle Corporation – Jersey (2016). Guía de usuario, versión 2.23.
16. Red Hat, Inc (2004). Documentación de referencia Hibernate 4.1.12. final.
17. Apache Software Foundation (2016). Documentación Log4j
18. The Apache Software Foundation (11 de abril de 2016). Documentación Apache Tomcat 7.
19. MySql (2016). Manual de referencia versión 5.5.
20. Developing RESTful APIs whit JAX-RS. Java Brains
21. The JQuery Foundation (2016), Documentación de AJAX.
22. ECMA INTERNATIONAL (Octubre 2013). The JSON Data Interchange Format
23. Refresh Data (2016). Documentación y ejemplos de HTML CSS y otros.
24. Bootstrap: http://getbootstrap.com/
25. Google Maps APIs documentation
26. Slick.js: http://kenwheeler.github.io/slick/
27. Sortable: https://jqueryui.com/sortable/
28. Draggable: https://jqueryui.com/draggable/
29. Droppable: https://jqueryui.com/droppable/
30. DataPicker: https://jqueryui.com/datepicker/
31. FontAwesome: http://fontawesome.io/
32. Moment.js http://momentjs.com/
33. Fermin Pitol (2014). Clustering, algoritmo de K.-Means.